A Large-Scale Mathematical Model of the Rat Pulmonary Circulation and the Effect of Chronic Hypoxia

Loolu Rafeeq
Marquette University

A LARGE-SCALE MATHEMATICAL MODEL OF THE RAT PULMONARY CIRCULATION AND THE EFFECT OF CHRONIC HYPOXIA

by

Loolu Rafeeq, B.E.

A Thesis submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Master of Science

Milwaukee, Wisconsin

August 2012

ABSTRACT
A LARGE-SCALE MATHEMATICAL MODEL OF THE RAT PULMONARY CIRCULATION AND THE EFFECT OF CHRONIC HYPOXIA

Loolu Rafeeq, B.E.

Marquette University, 2012

Mathematical models are useful for developing understanding of the behavior of complex biological systems. Recent work on a "physiome" project is aimed at using computational modeling to analyze integrative biological function by developing a simulation system for hypothesis testing (Borg & Hunter, 2003). To date, extensive work on cardiovascular, endocrine and nervous system models has been undertaken. Our objective here is to contribute to this effort by further developing a comprehensive integrative model of the pulmonary circulation.

A computational model of the dog pulmonary circulation was originally developed by Haworth et al. (Haworth S. T., 1996; Haworth, Linehan, Bronikowski, & Dawson, 1991). In this thesis, their work was extended to the rat pulmonary circulation. The rat model geometry is characterized by 18 orders of arteries and 19 orders of veins. The average distensibility (% increase in diameter over the undistended diameter) for the model arteries and veins are 2.8 %/mmHg and 1.6 %/mmHg. These arterial and venous trees are connected by a capillary sheet with an area of 0.123 cm^2. The model was validated and the calculated pressures, arterial-capillary-venous resistances, volumes, and compliances of the model agree well with the experimental estimates in the rat lung under zone 3 conditions. The model was used to evaluate the common structural hallmarks of pulmonary vascular remodeling as a result of exposure to chronic hypoxia (low inspired oxygen levels), such as the decrease in arterial and venous distensibility, reduction in capillary surface area and reduction in the number of small arteries.

Our results show that these factors are not alone sufficient to account for the reported increase in pulmonary arterial pressure in response to chronic hypoxia induced pulmonary hypertension. This extended model provides a graphical user interface for choosing parameters, simulation models, and numerical options for performing either steady state or dynamic simulations, and for displaying model simulation results. The results of this study demonstrate the potential utility of this model for furthering the understanding of the underling mechanisms of pulmonary hypertension, and the effects of other lung disorders on the pulmonary circulation.

ACKNOWLEDGMENTS

Loolu Rafeeq, B.E.

I sincerely thank my advisor Dr. Anne V. Clough who guided me through the entire process of this thesis work. I could not have completed this work without her constant support, insightful inputs and help with the editing. I thank her for being patient with me while I completed this work and for teaching me to pay attention to details. I owe my deepest gratitude to Dr. Steven T. Haworth for guiding me with the software design and model validation, helping me to gain enough background on the lung model, and providing answers to my several questions. I am extremely grateful to my committee member Dr. Said H. Audi for suggesting the thesis topic, for his valuable feedback, and for patiently guiding and helping me with the final edits. I thank Dr. Robert C. Molthen for providing the experimental data, clarifications, and insightful review comments.

A special thanks to Dr. Dean C. Jeutter, my academic advisor, for his motivation encouragement and guidance throughout my time at Marquette. I also thank my supervisor; Dr. Renee Theiss at Rehabilitation Institute of Chicago for being a role model and for her constant “you can do it” encouragements.

This thesis and my graduate studies at Marquette would not have been possible without the help, patience, love and support of my husband, Abdul Salim. I am extremely grateful to my mom, who took care of my daughter while I was finishing my academic career at Marquette University. I dedicate my thesis to my beloved kids, Suha Salim and the one ready to arrive anytime.

Last but not the least; I thank God almighty for giving me the strength and perseverance during this project and throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION AND AIMS

The primary goal of a mathematical model is to aid in describing the physical processes and behavior of a system using mathematical equations, often in a simplified manner (Tawhai, Clark, & Burrowes, 2011). Because of the complexity of living organisms, physiologic models are typically limited to a specific organ or even a specific biological process, rather than the entire system. In recent years, efforts have been made on the *Physiome Project*, whose objective is to develop comprehensive quantitative models of biological processes (Bassingthwaighte, 2000). These processes cover a wide range of systems, from subcellular organelles to whole organisms. The main goal of the *Physiome Project* is to use computational modeling to analyze integrative biological function, by developing a simulation system for hypothesis testing (Hunter & Borg, 2003). To date, extensive development of cardiovascular, endocrine and nervous system models has been achieved. The overall objective of this thesis is to contribute to the *Physiome Project* by constructing a comprehensive integrative model of the pulmonary circulation of the rat.

The pulmonary circulation carries oxygen-depleted blood from the right side of the heart to the lungs and delivers oxygenated blood back to the left side of the heart for its distribution to the systemic circulation. The primary function of the lung is to provide a robust environment for gas exchange, where within the capillaries, carbon dioxide diffuses from the blood into the alveoli, and oxygen diffuses out of the alveoli into the blood.

The main pulmonary artery delivers blood from the right ventricle to the lungs, where it bifurcates into smaller arteries forming a tree-like structure that terminates at the smallest arterioles. The arteriole has a strong muscular wall capable of closing the arteriole completely or allowing it to be dilated several fold, thus producing the capability of vastly altering blood flow to the capillaries in response to the need of the tissues (Guyton & Hall, 1996). The final generation of arterioles empties into the capillaries, where the exchange of gases occurs. Exchange of O_2 and CO_2 takes place between blood cells and alveoli through the process of diffusion. Following transit through the capillaries, the blood exits into the small venules, which converge into larger veins. This converging network of veins again forms a tree-like pattern coalescing to form the pulmonary vein, where the blood finally leaves the lung and empties into the left atrium of the heart. The blood is then pumped by the left ventricle of the heart through the aorta to the rest of the body. The pulmonary circulation is shown in **Figure 1.1**.

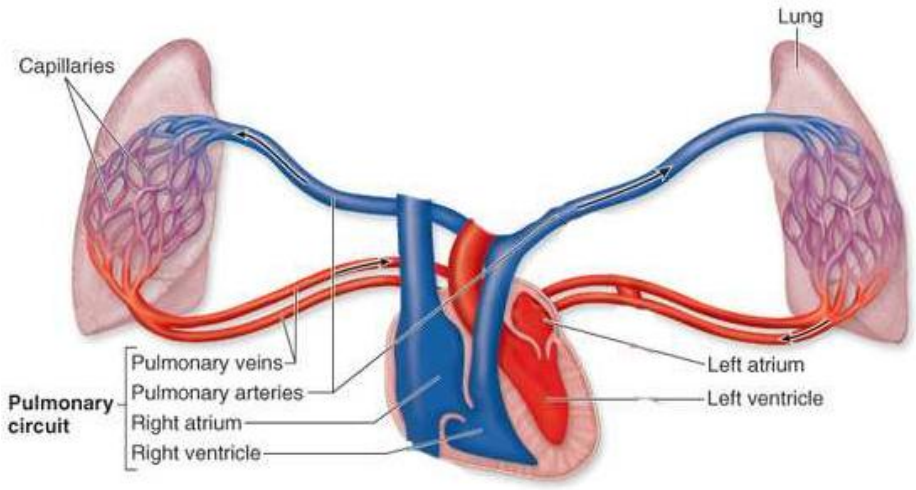


Figure 1.1: The pulmonary circulation. Blue indicates deoxygenated blood and red oxygenated blood (Iqbal, 2005).

Previously, Haworth et al. (Haworth, Linehan, Bronikowski, & Dawson, 1991) developed a large-scale model of the pulmonary circulation of the dog. Their model was based on the fundamental physical laws governing fluid dynamics and incorporated experimental data from the literature. This project aims to extend that work by developing a mathematical and computational model of the pulmonary circulation of a rat.

Over the years the laboratory rat has been used in many experimental studies which have added to our understanding of many biological relationships and other phenomena in health and medicine. There are similarities in pulmonary morphology, physiology, and lung maturation between rats and humans (Bolle, et al., 2008). For example, the morphological studies in lung growth from birth to adulthood between rats and humans show that the enlargement factor for lung volume was 23.4 in humans and 23.5 in rats and the estimate of pulmonary diffusion capacity for O_2 increased by a factor of 33 in both species (Bolle, et al., 2008). Rats reproduce rapidly, are inexpensive and easy to handle, and can be genetically manipulated at the molecular level. They can be selected from different strains to model various diseases with relatively low cost. For these and other reasons, the rat is a commonly used species for modeling and investigating the pulmonary circulation under physiologic and pathophysiologic conditions.

Pulmonary arterial hypertension (PAH) is a condition in which the blood pressure within the arteries of the lung is elevated. PAH is a significant public health problem and usually occurs along with other conditions such as blood vessel and heart diseases, lung diseases, sleep apnea or thyroid diseases. Between 2000 and 2002, 807,000 patients were

hospitalized with PAH and in 2002 alone, PAH led to 15,668 deaths and 260,000 hospital visits in United States (Hyduk, et al., 2005).

Pulmonary arterial hypertension is characterized by persistent vasoconstriction, narrowing and tightening of blood vessels and medial smooth muscle thickening of the pulmonary arteries (Jones & Reid, 1995). Eventually pulmonary vascular remodeling, i.e. a change in the structural and biomechanical architecture of the pulmonary vessels, occurs. The net result of these processes is increased pulmonary artery pressure and vascular resistance, which results in an increased workload for the right ventricle.

Numerous animal models of pulmonary hypertension are currently used to explore the biochemical mechanisms involved and to identify and evaluate new therapies (Zhao, 2011). The chronic hypoxia model, involving exposure to a low oxygen environment for extended periods of time, has been used in our laboratory as an experimental model of specific features of pulmonary hypertension (Molthen, Karau, & Dawson, 2004). In this model, the rat displays significant arterial remodeling consistent with PAH. Moreover, the biochemical and structural changes measured by scientific teams are predictable and reproducible (Zhao, 2011). However, the relative contributions of various biochemical, mechanical, and structural changes within the pulmonary circulation to the development and progression of pulmonary hypertension are still not fully understood. Thus, we propose to develop and use a mathematical model of the rat pulmonary circulation to interpret experimental data and to evaluate hypotheses related to the effects of chronic hypoxia in the rat model.

This extended model of the rat pulmonary circulation is then used to evaluate underlying factors involved in the increased pulmonary arterial pressure with exposure to chronic hypoxia. The final product of this pulmonary circulation model will include a user-friendly interface for performing simulations designed to comprehensively understand the impact of multiple process on the behavior of the rat pulmonary circulation.

Thus, the specific aims of this project are to:

1. Develop a computational model of the rat pulmonary circulation that can simulate both steady state (constant blood flow and tracheal pressure) and dynamic conditions (pulsatile blood flow and ventilation). The model will include i) a graphical user interface for choosing input parameters, model assumptions, and numerical options; and ii) an interactive user interface to visualize the output and provide options for exporting these results for further analysis.
2. Validate the computational model by comparing the model output with published experimental results under a variety of different physiological conditions; and
3. Use the model to test hypotheses regarding potential contributors to observed changes in pulmonary hemodynamics in the chronic hypoxia rat model.

2. LITERATURE REVIEW

2.1 Physiome Project

The *Physiome Project*, initially envisioned by Dr. James Bassingthwaigthe and now led by the International Union of Physiological Sciences (IUPS), is a program to archive and disseminate quantitative information and integrative models of the functional behavior of all scales of biological and physiological systems – from molecule to organism. More specifically, its goal is to use computational modeling to analyze integrative biological function so as to provide a system for hypothesis testing (Borg & Hunter, 2003). The name comes from "physio-" (life) and "-ome" (as a whole), and is defined as the "quantitative description of the functioning organism in normal and pathophysiological states" (Bassingthwaighte, 2000). To date, extensive cardiovascular, endocrine and nervous system models have been developed (Hunter, Crampin, & Nielsen, 2008)

2.2 Lung Physiome

The pulmonary research group at the University of Auckland Bioengineering Institute is currently developing anatomically- and biophysically- based computational models of the human pulmonary system under the auspices of the *Lung Physiome* (www.physiome.org.nz/lung) (Yin, Choi, Hoffman, Tawhai, & Lin, 2010). Their

efforts include implementation of models representing various pulmonary functions to study structure-function relationships (Burrowes & Tawhai, 2006; Tawhai, Nash, & Hoffman, 2006). They have derived computation-ready models of geometric structure to partner imaged subjects in a digital atlas of the human lung spanning several decades of life (Tawhai, et al., 2004; Burrowes, Hunter, & Tawhai, 2005; Lin, Tawhai, McLennan, & Hoffmann, 2007) (Tawhai, Hoffman, & Lin, 2009).

2.3 Other Pulmonary Hemodynamic models

Several previous modeling studies have investigated structure–function relationships in the arterial and venous networks of the lung. Most of these have reduced the complexity of the pulmonary vascular tree geometry by representing the arteries and veins as a symmetric tree (Parker, Cave, Ardell, Hamm, & Williams, 1997), as a self-similar fractal tree (Bennett, Goetzman, Milstein, & Pannu, 1996) or an average flow path via summary morphometric parameters (Dawson, et al., 1999). These early models represent only the average geometry of the branching structure because they were implemented to investigate the effects of large-scale changes in branching geometry (Burrowes, Swan, Warren, & Tawhai, 2008). Although these models were designed for use in a particular functional investigation, their results all suggested a strong dependence of the regional pulmonary blood flow distribution on the pulmonary vascular geometry. Burrowes et al. used pulmonary vascular models with anatomical detail to investigate the relative roles of the vascular branching structure and gravity on perfusion by comparing flow in symmetric and anatomical models, and in models with and without gravity (Burrowes & Tawhai, 2006). Solution of a one-dimensional form of the Navier–Stokes

equations including gravity gave predictions of spatially distributed blood pressure, vessel radius and blood flow. As a result of asymmetric branching structure, their model predicted a large amount of flow heterogeneity. When changing between prone and supine postures, the relative position of the main vessel inlet/outlet to the system played a large role in the relative blood flow distribution (Burrowes, Swan, Warren, & Tawhai, 2008). Regardless of the magnitude and direction of the gravity, the influence of branching geometry dominated in this model (Burrowes & Tawhai, 2006).

To accommodate for capillaries and the smallest arterioles and venules, a common model for blood flow within the microcirculation, known as 'sheet flow model', has been used. Because of the high density of pulmonary capillaries, flow is modeled as a continuous sheet bounded on either side by a compliant endothelium, and flowing between 'posts' of connective tissue (Fung & Sobin, 1969) rather than through discrete vessels. In this case, only the average value of hemodynamic variables (e.g., blood pressure, blood flow) over the entire capillary network can be determined; segment-to-segment variability in individual capillary pathways cannot be predicted. More recently, Dhadwal et al. and Huang et al. (Huang, Doerschuk, & Kamm, 2001; Dhadwal, Wiggs, Doerschuk, & Kamm, 1997) modeled blood flow through discrete tubules to represent the segmented structure of the capillary network. These early models used a simplified geometric structure consisting of a relatively small number of capillaries to represent the complex capillary network. The functional model of Huang et al. (Huang, Doerschuk, & Kamm, 2001) was extended by Burrowes et al. (Burrowes, Tawhai, & Hunter, 2004) by implementing their model equations in a more realistic representation of the capillary network geometry. This extended model involves a two- dimensional meshing method

generating a continuous network of capillaries over adjacent model alveoli in a single alveolar sac.

Multiple connection points were made between capillaries of surrounding alveoli and a single capillary sheet between the alveolar faces. This model was the first attempt to accurately show the relationship between the space filling-alveoli and the segmented geometry of the pulmonary circulation (Burrowes, Swan, Warren, & Tawhai, 2008).

The pulmonary physiology group at Zablocki VA Medical Center has studied the physiological, biochemical and histological responses to various lung diseases in a variety of species, including rats, mice and guinea pigs. A computational model of the pulmonary circulation for a dog was developed by Haworth (Haworth S. T., 1996), and is explained in detail in Chapter 3. Our objective here is to contribute to this effort by developing a comprehensive integrative model of the pulmonary circulation of the rat to study the hemodynamics of the pulmonary vasculature and the consequences of vascular remodeling.

2.4 Chronic Hypoxia and pulmonary vascular remodeling

Chronic exposure to low inspired oxygen levels (hypoxia) is a well-established animal model of pulmonary hypertension (Clough, Audi, Molthen, & Krenz, 2006). Researchers have unraveled various etiology components responsible for hemodynamic changes observed with chronic hypoxia. Chronic hypoxia elicits a variety of cardiopulmonary responses, including pulmonary hypertension (Hislop & Reid , 1976), increased vascular resistance, especially in the pulmonary circulation (Vanderpool, Kim, Molthen, & Chesler, 2010), right ventricular hypertrophy, polycythemia and structural

changes in the pulmonary vascular beds, collectively known as vascular remodeling (Zhao, 2011). Common structural hallmarks of pulmonary vascular remodeling include narrowing of the artery lumen, vessel wall thickening, and a decrease in the number of small arteries (Clough, Audi, Molthen, & Krenz, 2006). Other structural changes include abnormal extension of muscles into peripheral arteries where it is not normally present, increased wall thickness of the normally muscular arteries, and reduction artery number (Rabinovitch, Gamble, Nadas, Miettinen, & Reid, 1979).

In many rat models of pulmonary vascular remodeling, one of the first structural features observed is the appearance of neomuscularization of the small pulmonary arteries. This neomuscularization tends to increase pulmonary vascular resistance and decrease the number of parallel pathways, thereby diminishing excess capacitive volume in the vascular bed (Clough, Audi, Molthen, & Krenz, 2006). Vessel wall thickening is frequently associated with stiffer vessels as measured by vessel distensibility, the ability to distend in response to an increase in vascular pressure.

Molthen et al. demonstrated that rat exposure to hypoxia (10% O_2) for 21 days results in a 50% decrease in pulmonary arterial distensibility, a key feature of human pulmonary hypertension (Molthen, Karau, & Dawson, 2004). This decreased distensibility of the pulmonary arteries, associated with long-term exposure to hypoxia, may be responsible for the increase in pulmonary arterial pressure (Molthen, Karau, & Dawson, 2004). In addition to decreased distensibility, they reported increased blood hematocrit and right ventricular hypertrophy. Other experiments on rats exposed to chronic hypoxia also reported proximal arterial thickening and persistence of stiffening of the arteries (Vanderpool, Kim, Molthen, & Chesler, 2010). In another experiment,

Molthen et al. also reported a decrease in venous distensibility when rats were exposed to chronic hypoxia 10% O_2 for 21 days (Molthen, Gordon, Krenz, & Clough, 2007).

There is an ongoing debate regarding whether or not chronic hypoxia-induced pulmonary hypertension leads to loss of small peripheral arteries (Rabinovitch, Chesler, & Molthen, 2007). This loss could increases resistance by reducing parallel vascular pathways. Researchers have shown that in chronic hypoxia animal models, the ratio of pulmonary arterioles to pulmonary alveoli is reduced, suggesting loss of these blood vessels termed as 'rarefaction' or 'pruning' (Hislop & Reid, 1976; Rabinovitch et al. 1979; Jones & Reid, 1995; Partovian et al. 2000). Rabinovitch et al. characterized histologically the overall pattern of vascular remodeling in hypoxic rats: extension of smooth muscle into small previously non-muscular arteries, medial thickening in normally muscular arteries, and a decrease in the number of arteries that filled upon injection of a barium contrast agent was observed (Rabinovitch, Gamble, Nadas, Miettinen, & Reid, 1979). Hislop and Reid (Hislop & Reid, 1976) reported that in hypoxic rats, the microscopic counts of small arteries showed that vessels up to 200 μm external diameter were gradually "lost," reducing the ratio of arterial to alveolar number significantly by 14 days and no vestiges of these vessels were found with light microscopy. Hence, researchers have suggested that vascular structural changes, such as loss of small blood vessels (rarefaction), account for at least a portion of the increase in pulmonary arterial pressure (Zhao, 2011;Vanderpool, 2011).

Molthen et al. reported a decrease in capillary surface area of approximately 15% in rats exposed to chronic hypoxia (Molthen, Heinrich, Haworth, Krenz, & Gordon, 2004). Thus, it has been postulated that rarefaction is an important component for the

structural basis for hypoxic pulmonary hypertension. However, other studies of pulmonary hypertension in the rat, report that the ratio of pulmonary arterioles to alveoli was unaltered following chronic hypoxia exposure (Hopkins, 2002; Rabinovitch, 2007; Zhao 2011). Thus, the question of the impact of chronic hypoxia on small vessels remains open as a recent study reports the formation of new vessels (angiogenesis) in this lung injury model (Zhao, 2011). The proposed computational hemodynamic rat model will help to evaluate the impact of vessel distensibility and/or vessel rarefaction on pulmonary arterial pressure.

3. LUNG MODEL

3.1 Background and Overview

A computational implementation of the pulmonary circulation model was originally developed by Haworth (Haworth S. T., 1996). It was founded on fluid dynamic principles, including conservation of mass and momentum, which lead to a set of governing differential equations, known as the transmission line equations (Leung, Dumont, Sandor, & Potts, 2006; Palladino, Drzewiecki, & Noordergraaf, 2000; Haworth, Linehan, Bronikowski, & Dawson, 1991). Tree structures representing the branching geometry of the arteries and veins are used to represent a discretized set of ordered vessels consistent with experimental morphometric data. The capillary bed is integrated into the model using a sheet-post model (Fung & Sobin, 1969), and coupled together with the arterial and venous trees and the respiratory system, including respiration. Published data on the biomechanics of the lung are integrated as functional modules describing vessel distensibility, vessel interdependence, lung inflation, and blood rheology (Haworth S. T., 1996). The transmission line equations are then solved numerically at each node throughout the artery-capillary-vein system. The basic approach of this model has been previously validated for the dog pulmonary circulation.

The goal of this project is to develop a version of the above model for the rat pulmonary circulation. Section 0 describes the foundation of the model and assumptions used in developing the model. Section 3.3 describes the geometry of arteries, veins and capillaries of the rat, which is used as input to the model. Section 3.4 describes the

biomechanical and rheological properties of the pulmonary vasculature, which are incorporated as functional modules in the model. Section 3.5 describes the various model options, including occlusion, rarefaction and vasoconstriction. Section 3.6 describes the steady state and dynamic model implementation and their solutions that return the distribution of pressures and flows throughout the vascular segments of the lung, as a function of time.

Sections 3.3 to 3.6 include brief description of the models and equations that constitute the model, and are taken from the doctoral dissertation of Haworth (Haworth S. T., 1996).

3.2 Transmission Line Equations (TLE)

The foundation of the electrical analog model used in this thesis is a set of differential equations, referred to as the transmission line equations (Noordergraaf, 1969), which are derived starting with the physical laws of mass and momentum balance for flow through a blood vessel under the following assumptions (Milnor, 1989; Noordergraaf, 1969):

a. Blood vessels are cylindrical tubes with linearly elastic walls (Hookean material).

b. Blood is an incompressible (constant density), Newtonian (constant viscosity) fluid.

c. There is zero circumferential flow and the axial flow is independent of radial oscillations.

In the TLE equations, vascular resistance is represented by an electrical resistor (R), and vascular compliance is represented by an electrical compliance (C). The force

due to fluid inertia (fluid inertance) is equal to mass of the fluid multiplied by the derivative of the fluid flow rate with respect to time (acceleration of the fluid). This fluid inertia is represented in the TLE equations by an electrical inductor (L).

3.3 Vessel Geometry

3.3.1 Arteries and Veins

Detailed knowledge of the morphometry of the pulmonary arterial and venous trees is essential for developing a deterministic model of the pulmonary circulation. Morphometry is defined as a quantitative summary of the tree structure. Geometric data, i.e., diameters, lengths and number for arteries and veins of a given diameter/length, have been obtained from plastic corrosion casts of lungs of rat, cat, dog and human (Horsfeild, 1978). Moreover, data obtained from imaging methods are also becoming available (Tawhai, Clark, & Burrowes, 2011; Shingrani , Krenz, & Molthen, 2010). This extensive geometric vessel data is summarized by binning the vessels according to different ordering schemes (Suki, et al., 2003). The ordered data can then be displayed by graphing the log of the number of vessels of a given diameter or the corresponding length of the vessel segment versus the log of the mean diameter in each order as shown in **Figure 3.1** and **Figure 3.2**.

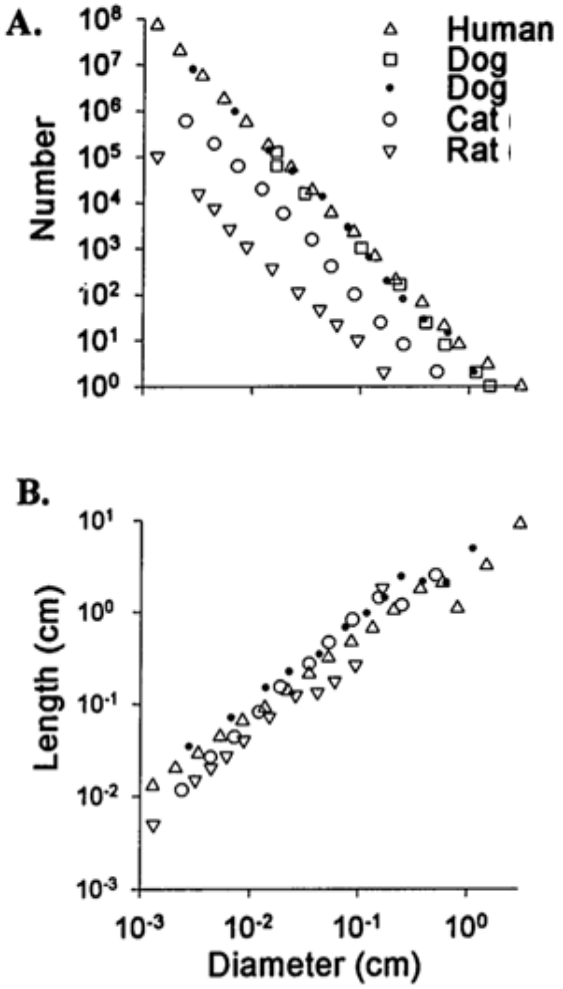


Figure 3.1: **A**: Log-log plot of the number of pulmonary arteries versus the mean vessel diameter within each order for the rat (Jiang, Kassab, & Fung, 1994), cat (Yen, Fung, & Bingham, Elasticity of small pulmonary arteries in the cat, 1980), dog (Gan & Yen, 1994; Miller, 1893) and human (Horsfield K., 1978; Singhal, Henderson, Horsfeild, Harding, & Cumming, 1973) lungs. For these data, the numbers of vessels were multiplied by two to permit comparison with the studies of the whole lung (Horsfield K., 1978; Singhal, Henderson, Horsfeild, Harding, & Cumming, 1973). **B**: Log- log plot of the length of pulmonary arteries versus the mean vessel diameter within each order for the rat (Jiang, Kassab, & Fung, 1994), cat (Yen, Fung, & Bingham, 1980), dog (Gan & Yen, 1994; Miller, 1893) and human (Horsfield K., 1978; Singhal, Henderson, Horsfeild, Harding, & Cumming, 1973) lungs. Reproduced from Haworth (Haworth S. T., 1996).

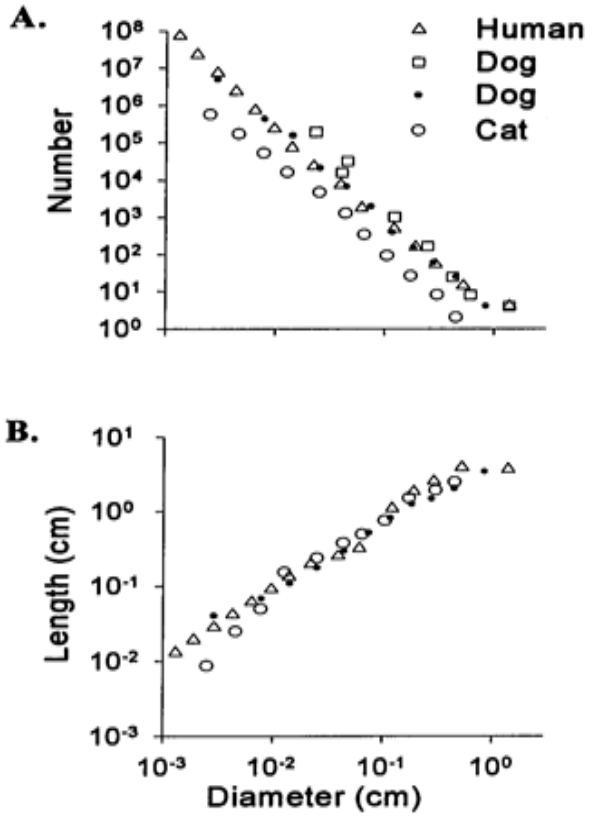


Figure 3.2: **A**: Log-log plot of the number of pulmonary venous vessels versus mean vessel diameter within each order for the cat (Yen, et al., 1983), dog (Gan, Tian, Yen, & Kassab, 1993) and human (Horsfield & Gordon, 1891) lungs. For these data, the numbers of vessels were multiplied by two to permit comparison with the studies of the whole lung. **B**: Log- log plot of the length of pulmonary venous vessels versus the mean vessel diameter within each order for the human (Gan, Tian, Yen, & Kassab, 1993), dog (Gan & Yen, 1994) and cat (Yen, et al., 1983) lungs. Reproduced from Haworth (Haworth S. T., 1996).

The log-log plots in **Figure 3.1** and **Figure 3.2** reveal nearly linear relationships between vessel number and vessel segmental length and mean diameter. Here the vessels are discretized into orders defined by their unstressed diameter. The equations describing the relationship between the number N of vessels with diameter D_j and/or length ℓ_j

which is consistent with the data are represented by the power law functions given in Equations (3.1) and (3.2), where subscript *j* is the order number.

$$N(D_j) = a_1 D_j^{-\beta_1} = N_1 \left(\frac{D_j}{D_1} \right)^{-\beta_1} \quad 3.1$$

$$\ell(D_j) = a_2 D_j^{\beta_2} = \ell_1 \left(\frac{D_j}{D_1} \right)^{\beta_2} \quad 3.2$$

The parameters β_1 and β_2 represent the slopes determined from the linear regression of the log number or length diameter measurements, and are the key descriptors of the morphometry of the vessel trees. The parameters a_1 and a_2 are the intercepts from the linear regression plots and correspond to the scaling factors to accommodate the size of the tree (Haworth, Linehan, Bronikowski, & Dawson, 1991). Even though, there are differences in the branching patterns at the macroscopic level, these graphs are remarkably similar for all the species studied with the most notable difference being the intercept values reflecting the difference in lung size (Suki, et al., 2003).

Also, the ratio of successive vessel diameters, *r*, is defined as

$$r = \frac{D_{j-1}}{D_j} = \left(\frac{D_1}{D_n} \right)^{\frac{1}{n-1}}. \quad 3.3$$

Here, *n* is the total number of orders from the largest vessel diameter ($j = 1$) to the smallest non-capillary vessels *n*. For a homogeneous tree structure, the cumulative arterial and venous vascular volume from the inlet artery or the outlet vein to the end of the order *j* is given by $V(j)$.

$$V(j) = \sum_{j=1}^{n} V_j \quad ; \quad V_j = \frac{\pi D_j^2 \ell_j N_j}{4}. \quad 3.4$$

We are using the key parameters (β_1, β_2, a_1, a_2) and volume information from these experiments to establish the geometry of the pulmonary arteries and veins in rats.

Establishing the rat arterial and venous geometry:

The model geometry for arteries and veins are established using the continuum model by Krenz (Krenz, Linehan, & Dawson, 1992). This model gives the optimized parameter values for the initial arterial and venous diameters $D(0)$ and the a_2 based on a set of geometric parameter inputs in relevant ranges obtained from the experiments. We again followed the work of Haworth (Haworth S. T., 1996) and used these optimized parameter values to determine the total number of vessel orders, n, for arteries and veins. The geometric parameter inputs and their values, which specify the vessel length, number and diameter for the lobar arterial and venous trees for the rat model, are shown in **Table 3.1.**

Input Parameter (Units)	Parameter description	Arterial Tree	Venous Tree	Reference
β_1	Slope of log N vs log D	2.5	2.5	(Suki, et al., 2003; Jiang, Kassab, & Fung, 1994)
β_2	Slope of log l vs. log D	1	1	(Jiang, Kassab, & Fung, 1994)
Pa_i; Pv_o (cm-H_20)	Inlet arterial pressure; Outlet venous pressure	8	0.1	(Molthen, Wietholt, Haworth, & Dawson, 2004)
R_L (*Ra, Rv*) (cm-H_20sec/ml)	Lobar resistance	2	5	(Intengan, Thibault, Li, & Schiffrin, 1999)
V_L (*Va, Vv)* (ml)	Lobar volume	0.31	0.31	(Molthen, Wietholt, Haworth, &

				Dawson, 2004)
α (%/mmHg)	Distensibility coefficient	2.8	1.6	(Molthen, Karau, & Dawson, 2004; Molthen, Gordon, Krenz, & Clough, 2007)
Υ (ratio)	D_{MAX} / $D(0)$	2	2	(Krenz, 1992)
Dterm (μm)	Terminal diameter	20	20	(Jiang et al., 1994)
B (ratio)	Branching ratio	2	2	(Jiang et al., 1994)
Hct (ratio)	Blood hematocrit	0.45	0.45	(Gwenda R. Barber, 1982)
$D(0)$ (cm)	Starting value for $D(0)$	0.2	0.2	(Haworth S. T., 1996)
a_2 initial	Starting value for a_2	6	3	(Haworth S. T., 1996)

Table 3.1: Geometric parameter inputs for the continuum model.

Parameter Optimization:

One requirement of the model is that the total vascular resistance is within the normal range for a rat, 19.62 - 31.2 cm-H_20 sec ml^{-1} (Hillyard, Anderson, & Raj, 1991). By using the inputs from **Table 3.1**, the optimized values of $D(0)$ and a_2 for arteries and veins were calculated utilizing the $R(j)$ and $V(j)$ cumulative volume functions given by equations 3.5 and 3.7 below (Krenz, 2003). The cumulative hemodynamic resistance $R(j)$, from the inlet artery to the end of the jth vessel order is expressed in terms of the resistance of the order 1 artery as

$$R(j) = \sum_{j=1}^{n} R_j = \sum_{j=1}^{n} \left(\frac{R_1 \mu_{a,j}}{\mu_{a,1}} \left[B^{\frac{(-\beta_1 - \beta_2 + 4)}{\beta_1}} \right]^{j-2} \right) \qquad 3.5$$

where

$$R_1 = \frac{128\mu_{a,1}\ell_1}{\pi N_1 D_1^4} = \frac{128\mu_{a,1}a_2 D_1^{\beta_2}}{\pi a_1 D_1^{(4-\beta_2)}}. \quad 3.6$$

The apparent viscosity, μ_a was calculated using the Kiani/ Hudetz model described in section 3.4.3. The cumulative vascular volume from order 1 to the *n*th order, is represented as

$$V(j) = \sum_{j=1}^{n} V_j = \sum_{j=1}^{n} \left(V_1 \left[B^{\frac{(-\beta_1 - \beta_2 - 2)}{\beta_1}} \right]^{j-1} \right) \quad 3.7$$

where

$$V_1 = \frac{\pi D_1^2 \ell_1 N_1}{4} = \frac{\pi D_1^2}{4} \; a_2 D_1^{\beta_2} a_1 D_1^{-\beta_1}. \quad 3.8$$

To begin, the iterative process of optimizing for the initial arterial and venous diameters, the starting values for $D(0)$ and a_2 were set equal to the values given in **Table 3.1**. To estimate values for $D(0)$ and a_2, the MATLAB optimization function 'FMINCON' which employs Levenberg Marquardt algorithm was used to find the parameter variables that give the best fit (minimum normalized orthogonal residual sum of squared differences) between $R(j)$ and $V(j)$ equations (3.5) and (3.7) and the input R_L and V_L (Haworth S. T., 1996). Using the initial estimate for $D(0)$, the total number of orders n is determined from Equation 3.3. The resulting optimal parameters are given in **Table 3.2**.

Optimized Parameters	Arterial tree	Venous tree
$D(0)$ (cm)	0.2335	0.1895
a_2	1.1414	1.08
N	18	19

Table 3.2: Parameter estimates the initial arterial and venous tree diameter $D(0)$, a_2 and the number of total orders n.

Based on the parameter estimates, the rat lung geometry consists of 18 orders of arteries and 19 orders of venous vessels. In the model lung, the main pulmonary artery is assigned order 1, the right and left branches are order 2, and lobar arteries assigned order 3, etc. Each order is distinguished by a mean vessel diameter and length. The ordering is homogeneous; so that flow through vessels of order $n+1$ is divided equally among the vessels of order n and all the vessels in order n have the same dimensions (Haworth S. T., 1996). The smallest arteries and veins are joined by a capillary sheet.

Figure 3.3 and **Figure 3.4** show the logarithmic plot of the lung model dimensions relating the number and their diameters and the length of the vessels and their diameters for the arterial and venous trees. The model geometry for individual orders is shown as the connected solid dots.

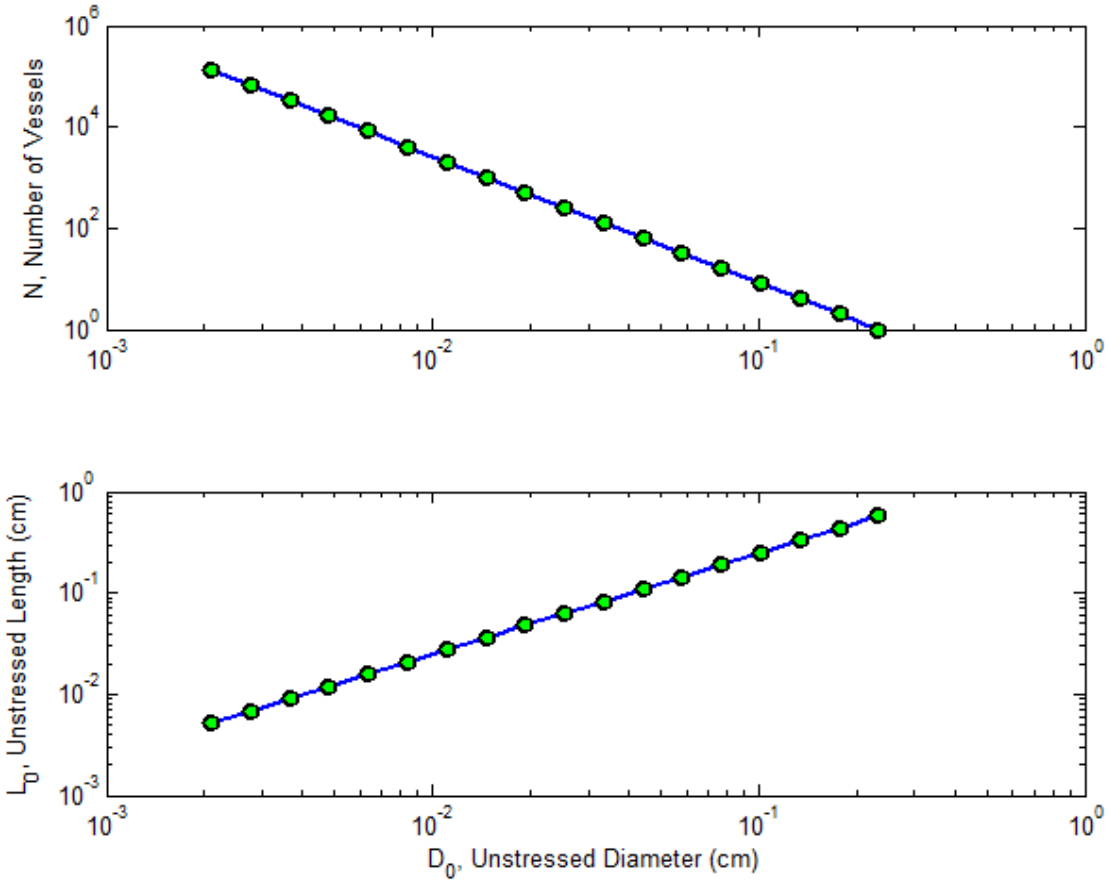


Figure 3.3: Log- log plot of the number of arteries versus their diameter within each order for the lung model (Top). Log–log plot of the length of each arterial segment versus diameter within each order for the lung model (Bottom).

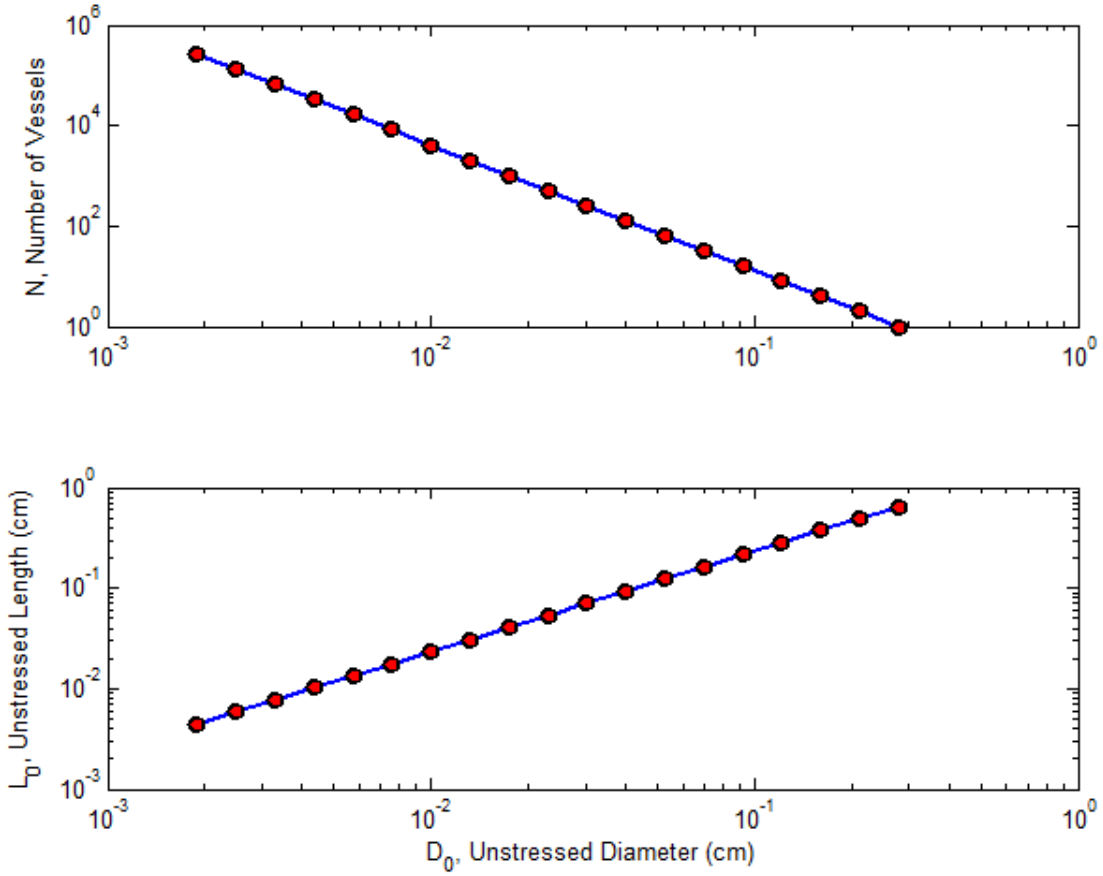


Figure 3.4: Log- log plot of the number of veins versus corresponding diameter within each order for the lung model (Top). Log –log plot of the length of venous segments versus the diameter within each order (Bottom).

Appendix 1(a) presents the input rat morphometry file for the model, including the values of N, D, L, viscosity (μ), vessel distensibility (α) and $\Upsilon(D_{MAX}/D(0))$ for arteries and veins of each order.

3.3.2 Capillaries

In 1969 Fung and Sobin proposed that the topology of the pulmonary capillary blood vessel network is not tree-like, but instead sheet-like (Fung & Sobin, 1969). The

microstructure of the pulmonary capillary system is a very dense system of tubules, but microvascular blood flow is commonly modeled as flat sheet flow between two membranes supported by equally spaced "posts". This plane is then divided into a network of hexagons, with a circular post at the center of each hexagon.

This sheet geometry is defined by the following parameters: sheet thickness (h), which is the distance between the roof and floor of the sheet, length (l_c), which is the length between the arterial entrance and venous exit, a width dimension, w, perpendicular to the flow and the diameter of the intermittent posts, ε, as shown in **Figure 3.5** (Haworth S. T., 1996).

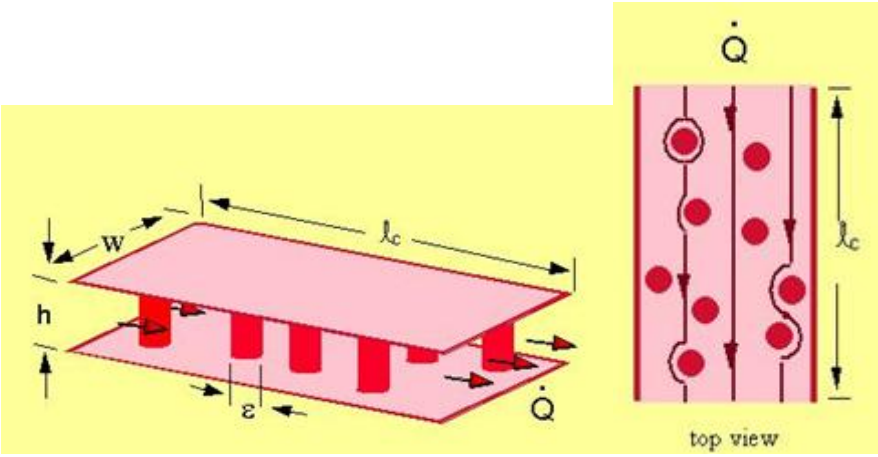


Figure 3.5: Capillary sheet representation from the side view (left panel) and top view (right panel) (Haworth S. T., 1996).

The solid lines represent examples of paths available to blood cells passing through the sheet (Haworth S. T., 1996).

Another property is *VSTR*, or vascular-space tissue ratio, which is the ratio of the volume of lumen space to the total space between the sheets (lumen and posts). The equation representing the *VSTR* is given by;

$$VSTR = 1 - \frac{n_p \pi \varepsilon^2}{4A} \quad 3.9$$

where n_p represents the number of posts and A is the sheet area (Fung & Sobin, 1969). The capillary blood volume is then calculated by the equation,

$$V_c = Ah\,VSTR \;;\; A = wl_c \quad 3.10$$

One advantage of assuming the sheet-post model for the capillary system is the computational advantage of computing flow through a single "sheet" versus each individual capillary.

Establishing capillary sheet geometry:

The geometric parameters specifying the capillary sheet dimensions in the model are *h, lc*, sheet area (*A*), and *VSTR*. The values for these parameters were obtained from the literature using other species (Kent E.Pinkerton, 1992) and are shown in **Table 3.3**.

Capillary sheet parameters	Value	Reference
Sheet height *h*(*0*)(μm)	5.63	(Haworth S. T., 1996)
Sheet length (l_c)(μm)	2.05	(Ramakrishna, 2009)
Sheet area (*A*)(cm^2)	0.123	(Fung & Sobin, 1972)
VSTR (Ratio)	0.743	(Maloney, 1969)

Table 3.3: The table shows the model capillary sheet parameters and their references.

Capillary sheet geometry in zone 2:

Various relative pressure conditions within the lung have come to be known as "zone" conditions. The different zones in the lung are shown in **Figure 3.6.** In the top region (Zone 1), the alveolar pressure (*PA*) is greater than the arterial pressure (*Pa*). In this case, the vessels are collapsed and there is no blood flow (Levitzky, 2006). Zone 2 is the condition where the pulmonary arterial pressure, *Pa* is greater than the alveolar

pressure, *PA*, which in turn is greater than the venous pressure *Pv*, i.e, *Pa* > *PA*> *Pv*. Under these conditions *PA* is greater than the intravascular pressure (*P*) somewhere within the pulmonary vascular bed, leading to the potential for hemodynamic resistance to increase since this resistance is dependent on the *PA* - *Pv* difference (Haworth S. T., 1996).

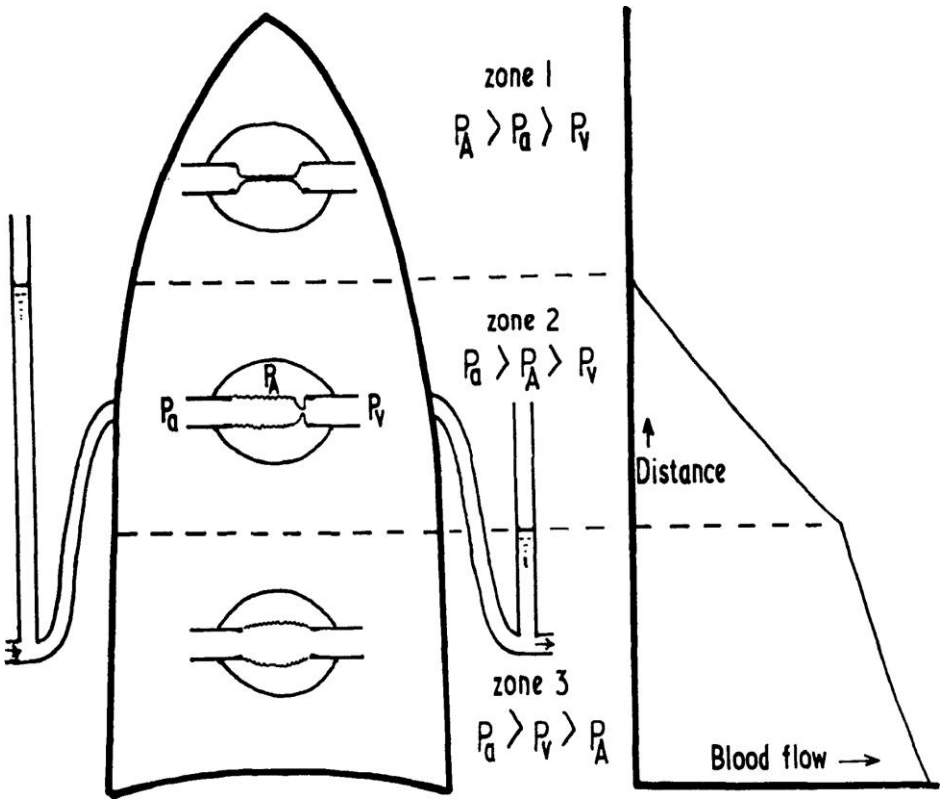


Figure 3.6: Left: The diagram representing the three zones of lung (zones 1-3). Middle: Section describes the relationship of Pa, PA and Pv for each of the 3 zones. Right: Graph showing the distance up the lung versus blood flow .Reproduced from Levitzky and West et.al (Levitzky, 2006; West, Dollery, & Naimark, 1964)

In our work, the so called "finite-thickness" model of the capillary sheet is used to represent zone 2 conditions. In this finite-thickness model, the thickness of the sheet, *h*, narrows only to a minimal thickness *hs* > 0. This approach was developed by Fry (Fry, Thomas, & Greenfeild, 1980) and later extended by Dawson et al. (Dawson, Rickaby, &

Linehan, 1986). This model accounts for both the volume and resistance changes that occur during changes in (*PA* - *Pv*) under zone 2 conditions. **Figure 3.7** is a diagrammatic representation of the finite-thickness model.

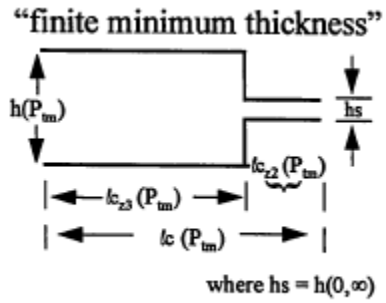


Figure 3.7: Finite-thickness model to account for the zone 2 pressure flow relationship. Reproduced from Haworth (Haworth S. T., 1996).

The length of the portion of the capillary sheet length in zone 2 is given by

$$l_{cz2} = \left[\frac{PA - P}{Q} \frac{Ahs^3 VSTR}{12\mu_a f} \right]^{1/2} \quad 3.11$$

while the length of the portion in zone 3, is

$$l_{cz3} = l_c - l_{cz2}. \quad 3.12$$

Similarly the area of the capillary sheet associated with lc_{z2} and lc_{z3} are

$$A_{z2} = l_{cz2} w \quad 3.13$$

$$A_{z3} = A\left(1 - l_{cz2}/l_{cz3}\right). \quad 3.14$$

3.4 Biomechanics

The biomechanical properties of blood vessels are developed as functional modules in the model, based upon extant published experimental results. There are different modules for vessel distensibility, vessel interdependence and blood rheology, as explained in the sections below. The parameter values from individual functional modules are given in Appendix 1(b).

3.4.1 Vessel Distensibility

Distensibility of the blood vessels is defined as the change in the vessel diameter due to a corresponding change in transmural pressure (P_{tm}), the pressure difference between intravascular (P) and the extra-vascular pressure (Px). There are two model options to calculate vessel diameter as a function of transmural pressure $D(P_{tm})$ in artery and vein segments: linear (Zhuang, Fung, & Yen, 1983) and nonlinear (Linehan, F.deMora, Bronikowski, & Dawson, 1988). The linear model is represented as

$$D(P_{tm})_j = D(0)_j[1 + \alpha P_{tm}] \qquad 3.15$$

whereas the nonlinear model (Haworth S. T., 1996) is represented as

$$D(P_{tm})_j = D(0)_j\left[\gamma - (\gamma - 1)e^{\frac{-\alpha P_{tm}}{\gamma - 1}}\right] \qquad 3.16$$

where, $D(0)_j$ represents the unstressed vessel diameter (at $P_{tm} = 0$) and γ is the ratio of maximum vessel diameter at high P_{tm} to $D(0)$. The value of α and γ are obtained from a nonlinear regression fit of the diameter versus pressure data for arteries and veins (Molthen, Karau, & Dawson, 2004) and are given in Appendix 1(b).

In the case of capillaries, the sheet thickness depends on both transpulmonary pressure (P_{tp}) and transmural pressure (P_{tm}), where, transpulmonary pressure (P_{tp}) is the pressure difference between airway (PA) and pleural pressure (Ppl) (Haworth S. T., 1996). Pleural pressure is the pressure surrounding the lung in the pleural space. There are two models that describe the mean capillary sheet thickness as a function of P_{tp} and P_{tm}. The linear model (Fung & Sobin, 1969) is represented by,

$$h(P_{tm}, P_{tp}) = [h(0,0) - (-\alpha * P_{tm})/(\gamma - 1)] \quad 3.17$$

and the nonlinear model (Glazier, Hughes, Maloney, & West, 1969) by,

$$h(P_{tm}, P_{tp}) = [h(0,0) - h(0,\infty)]e^{k_1 P_{tp}} + h(0,\infty)\left[\gamma - (\gamma - 1)e^{\frac{-\alpha P_{tm}}{(\gamma-1)}}\right]. \quad 3.18$$

where $h(0,\infty)$ represents the capillary sheet thickness when $P_{tm} = 0$ and P_{tp} is large and $h(0,0)$ is the capillary sheet thickness when $P_{tm} = 0$ and $P_{tp} = 0$. γ is the ratio of maximum thickness at large P_{tm} to $h(0,P_{tp})$ and α is the sheet distensibility coefficient. The capillary sheet distensibility value is within the same order of magnitude as the arteries and veins (Krenz & Dawson, 2003).

3.4.2 Vessel Interdependence

This section describes the various models for calculating the perivascular pressure, arterial and venous vessel length and volume, capillary sheet length, width, post diameter and volume. Section (i) describes the models for calculating the perivascular pressure which counteracts the expansion or contraction of the vessel lumen. The effect of lung volume on the intraparenchymal arteries and veins are described in Section (ii).

The capillary sheet length, width, post diameter as a function of volume are described in Sections (iii) and (iv).

i. Perivascular pressure

There are different models to describe the pressure that counteracts the expansion or contraction of the vessel lumen. For the intrapulmonary extra-alveolar vessels, P_{tp} has been found to depend on the lumenal pressure (P) and pleural pressure (Ppl), where P_{tp} is PA- Ppl. Several models have been suggested that describes the pressure that counteracts the expansion or contraction of the vessel lumen (Haworth S. T., 1996). The reported models for calculating the perivascular pressure ($\ddot{P}_x$) in arteries and veins are shown in equations 3.19, 3.20 and 3.21. These empirical expressions characterize the relationships between perivascular pressure ($\ddot{P}_x$), lumenal pressure (P) and transpulmonary pressure (P_{tp}). The equations to calculate the perivascular pressure for arterial and venous trees are shown below. The simplest equation fits the data from Albert et.al (Albert, Lamm, Rickaby, & al-Tinawi, 1993), where the perivascular pressure depends only on the transpulmonary pressure. Bshouty and Younes used the six parameter equation to express the relationship (Bshouty & Younes, 1990). A four parameter model was proposed as an alternative to the six parameter model (Haworth S. T., 1996) which fits the data from Smith and Mitzner (Smith & Mitzner, 1980).

Albert Model:

$$\ddot{P}_x = -P_{tp} \qquad 3.19$$

Bshouty and Younes Model:

$$P_x = (k_1 + k_2 P_{tp}) + (k_3 + k_4 P_{tp}) e^{[(k_5 + k_6 P_{tp})(P - P_{pl})]} \quad 3.20$$

Haworth and Smith/Mitzner Model:

$$P_x = [(P - P_{pl}) k_3 + k_4] P_{tp} - [(P - P_{pl}) k_3 + k_4] k_2 + k_1 \quad 3.21$$

In the case of capillaries, the capillary sheet perivascular pressure (P_x) is equal to the negative of *PA* (Haworth S. T., 1996).

$$P_x = -PA \quad 3.22$$

ii. Vessel Length and Volume

The effect of lung volume on the intraparenchymal arteries and veins are described in this section. The following representations describe the effect of lung volume (*V*) on the intraparenchymal artery and vein lengths. There are two models for representing the relationship between the vessel length (ℓ) in arteries and veins and the lung volume: the isotropic and Smith/Mitzner models (Smith & Mitzner, 1980; Haworth S. T., 1996). The isotropic model is represented by the equation

$$\frac{\ell_j}{\ell_{j,ref}} = \left[\frac{V}{V_{ref}} \right]^{\frac{1}{3}} \quad 3.23$$

where V_{ref} is the deflated (or end expiratory) volume including the volume of the tissue and trapped air. The subscript *j* represents the order number from the largest artery or vein diameter (*j* =1) to the smallest arterioles or venules (*j* = *n*). The Smith/Mitzner model for representing the vessel length and volume relationship is represented by the equation

$$\frac{\ell_j}{\ell_{j,ref}} = \left[\frac{V}{V_{ref}} + k_1 \exp\left(k_2 \frac{V}{V_{ref}}\right)\right]^{\frac{1}{3}} \quad 3.24$$

where ℓ_{ref} is the vessel length at V_{ref}.

iii. Capillary sheet length and width

This module calculates the length and width of the capillary sheet as a function of volume. Again, there is an isotropic and a non-linear (Smith/Mitzner) model (Haworth S. T., 1996; Smith & Mitzner, 1980). In the isotropic model, the sheet length (ℓc) and width (*wc*) are represented by the equation;

$$\frac{\ell c_j}{\ell c_{j,ref}} = \left[\frac{V}{V_{ref}}\right]^{\frac{1}{3}} \quad 3.25$$

$$\frac{wc_j}{wc_{j,ref}} = \left[\frac{V}{V_{ref}}\right]^{\frac{1}{3}} \quad 3.26$$

where wc_{ref} and lc_{ref} are the vessel width and length at the reference volume V_{ref}. For the Smith/Mitzner model, the length and width of the capillary sheet are represented as

$$\frac{\ell c_j}{\ell c_{j,ref}} = \left[\frac{V}{V_{ref}} + k_1 \exp\left(k_2 \frac{V}{V_{ref}}\right)\right]^{\frac{1}{3}} \quad 3.27$$

$$\frac{wc_j}{wc_{j,ref}} = \left[\frac{V}{V_{ref}} + k_1 \exp\left(k_2 \frac{V}{V_{ref}}\right)\right]^{\frac{1}{3}} \quad 3.28$$

iv. Capillary post diameter and Volume

This module solves for the diameter of posts (εc) in the sheet post capillary bed as a function of volume. There are three model options: isotropic, Gilbased and constant volume models (Haworth S. T., 1996). The isotropic model is represented by

$$\frac{\varepsilon c_j}{\varepsilon c_{j,ref}} = \left[\left(\frac{V}{V_{ref}}\right)\right]^{\frac{2}{3}}, \quad 3.29$$

the Gilbased model by

$$\frac{\varepsilon c}{\varepsilon c_{ref}} = \left[(1-k_1)\frac{V}{V_{ref}}\right]^{k2+k1}, \quad 3.30$$

and the constant volume model by

$$\frac{\varepsilon c}{\varepsilon c_{ref}} = \left[\left(\frac{h_{c0\_FRC}}{h_c}\right)\right]^{\frac{1}{2}}, \quad 3.31$$

where εc_{ref} represents the post diameter at V_{ref}.

v. Geometric Friction Factor:

The calculation of the pressure drop across the capillary sheet includes a non-dimensional geometric friction factor, *f*, defined by Yen and Fung (Yen & Fung, 1973) that describes the relationship between *VSTR*, *h*, *ε,* and *f* where, *ε* represents the diameter of the intermittent posts. This is represented by

$$f = (k1\ VSTR + k2)\exp\left[(VSTR\ k3 + k4)\left(\frac{h}{\varepsilon}\right)\right] \quad 3.32$$

3.4.3 Blood Rheology

Since blood does not behave as a continuum flow in vessels with diameters below 300 microns (Fahraeus Lindquist effect), the viscosity of blood depends on the size and

shape of the vessel and is characterized by 'apparent viscosity' (μ_a), which is a complex function of several parameters such as the blood hematocrit (Dhawal, 1993). Blood hematocrit (*Hct*) is defined as the blood cell volume per unit blood volume. There are two modules that represent the vessel diameter dependency of blood viscosity: a Kiani and Hudetz model (Kiani & Hudetz, 1991) and a Linehan model (Linehan, Haworth, Nelin, Krenz, & Dawson, 1992).

a. Kiani/ Hudetz model

This model is obtained from the superposition of two representations of the effect of diameter on viscosity. The representation of apparent blood viscosity is given here by

$$\mu_a = \mu_p \left[1-\left(1-\frac{\mu_p}{\mu_c}\right)\left(1-\frac{2\delta}{D(P_{tm})_j}\right)^4\right]^{-1}\left[1-\left(\frac{D_{\min}}{D(P_{tm})_j}\right)^4\right]^{-1} \quad 3.33$$

where, D_{min} is the observed minimum diameter of a single blood cell in a vessel and μ_c is the blood viscosity in large vessels (>300 microns) and is expressed as

$$\mu_c = \mu_p \exp\left(k_1 + Hct_f k_2\right) \quad 3.34$$

$$\delta = k_1 - k_2 Hct_f \quad 3.35$$

where μ_p is the apparent viscosity of plasma, δ represents the marginal plasma thickness layer and Hct_f is the feed hematocrit.

b. Linehan model

The Linehan model for apparent viscosity and *Hct* in the lungs is given by

$$\mu_a = \mu_p \exp\left(k_1\ Hct(D)\right) \quad 3.36$$

$$\frac{Hct(D)}{Hct_f} = k_1 \exp\left(-k_2 D\right) + \left(\frac{k_4 D}{k_3 + D}\right) \quad 3.37$$

This model assumes that viscosity depends only on hematocrit.

3.5 Model Options

In order to test and simulate different disease conditions, different model options are incorporated in this large scale model. The effect of acute hypoxia can be modeled using the vasoconstriction options described in Section **3.5.1**. The arterial and venous occlusion studies can be modeled using vascular occlusion models described in Section **3.5.2**. The effect of arterial rarefaction can be tested using the rarefaction option described in Section **3.5.3**. For the dynamic model, two different inputs of cardiac output can be chosen as in Section **3.5.4**.

3.5.1 Vasoconstriction

The hemodynamic responses of various vasoactive agents can be simulated using the lung model. Model simulation for arterial vasoactive stimuli, including serotonin (5-HT) and hypoxia, and venous vasoactive stimuli are described under sections **3.5.1.1** and **3.5.1.2**, respectively. These simulations were previously used as part of the validation of the dog lung model (Haworth S. T., 1996). Similarly, simulation of these experimental conditions is used in this study as part of the validation of the rat lung model.

3.5.1.1 Arterial vasoconstriction

Arterial vasoactive stimuli include acute hypoxia, for example, short-term inhalation of 10% O_2, and administration of serotonin (5-HT) (al-Tinawi, Krenz, Rickaby, Linehan, & Dawson, 1994). To simulate the hypoxic response in the model, the attempt was to reproduce the paradoxical effect of the large artery diameters increasing

and the small artery diameters decreasing as reported previously (al-Tinawi, Krenz, Rickaby, Linehan, & Dawson, 1994; Haworth S. T., 1996). This paradoxical effect is simulated in the model by changing the two morphometric parameters, β_1 and β_2. In order to increase the contribution of the vascular resistance of the small diameter vessels, the slope of the log *N* versus *D* curve is shifted and rotated (Haworth S. T., 1996). The functional form corresponding to vessel diameter changes during hypoxia, $D(0)_{j,H,}$ was expressed as a function of $D(0)_{j,C}$, $\beta_{1,C}$ and $\beta_{1,H}$ as

$$D(0)_{j,H} = D(0)_{1,c}\left[\frac{D(0)_{1,C}^{\,1-\frac{\beta_{1,C}}{\beta_{1,H}}}}{D(0)_{j,C}^{\,1-\frac{\beta_{1,C}}{\beta_{1,H}}}}\right] \qquad 3.58$$

where subscripts *C* and *H* represent control and hypoxic conditions. In this model, artery vessel lengths are not affected by constriction of vessel diameter (Haworth S. T., 1996). Then $\beta_{2,H}$ is determined from the log ℓ versus log *D* graph as

$$\beta_{2,H} = \frac{\beta_{2,C}\beta_{1,H}}{\beta_{1,C}} \qquad 3.59$$

The resulting log-log graphs of ℓ versus $D(0)$ are shown in for the range in $\beta_{1,H}$ from 2.4 to 2.2, where subscript *a* represents arteries and *j* the vessel order.

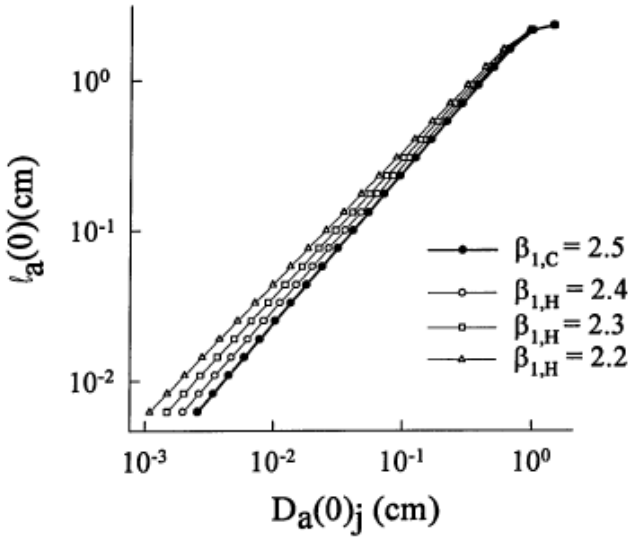


Figure 3.8: Log-log plot of the model simulated hypoxic arterial vessel lengths, $l_a(0)$, and diameters, $D_a(0)$ for various simulated hypoxia conditions are shown where $\beta_{1,H}$ ranges from 2.4 to 2.2 (Haworth S. T., 1996).

In the model simulation response of 5-HT (serotonin), all arteries are constricted by the same fraction of their original diameter (al-Tinawi, Krenz, Rickaby, Linehan, & Dawson, 1994). This results in a parallel shift of the slope of the artery log $N_{a,j}$ versus log $D_a(0)_j$ data curve as shown in **Figure 3.9**.

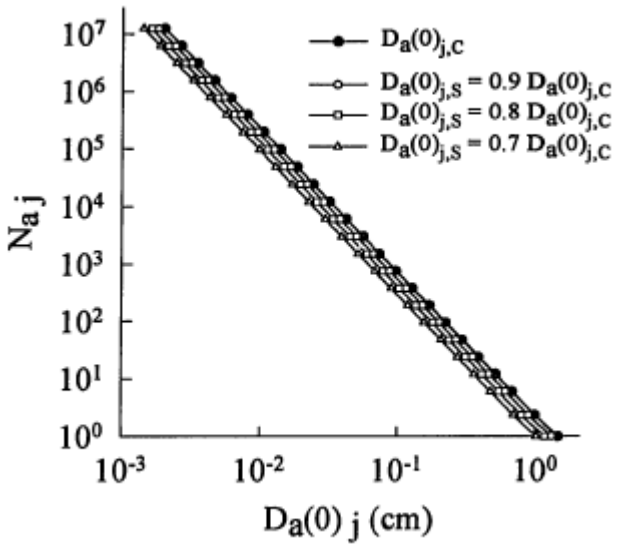


Figure 3.9: A log – log plot of artery numbers for given order, $N_{a,j}$ versus artery diameters $D_a(0)_j$, for Control ,C (solid circles) and for various degrees of arterial constriction (open symbols) ranging from 0.90 to 0.70 of $D_a(0)_j$. For each model, serotonin simulation, β_1 was equal to 2.5 (Haworth S. T., 1996).

3.5.1.2 Venous vasoconstriction

A vasoactive model of venous constriction is incorporated in order to simulate the effect of administration of histamine (Haddy, 1960). In simulating the response of histamine, the diameters of the veins are reduced in a manner similar to the serotonin simulations (Haworth S. T., 1996). Again, this results in a parallel shift in the slope of the log $N_{v,j}$ versus log $D_v(0)_j$ for the veins.

3.5.2 Vascular Occlusion

Occlusion is the process of rapidly stopping blood flow within the pulmonary artery (arterial occlusion) or the pulmonary vein (venous occlusion) or both (double occlusion). The transient vascular pressure in a lobar artery and vein, following a venous,

arterial or both arterial and venous occlusion, provides information about the longitudinal distribution of the vascular resistance relative to vascular compliance within the lung vasculature (Haworth S. T., 1996). The occlusion model can be explained using the simple electrical analog model representation of the pulmonary vasculature as shown in **Figure 3.10**.

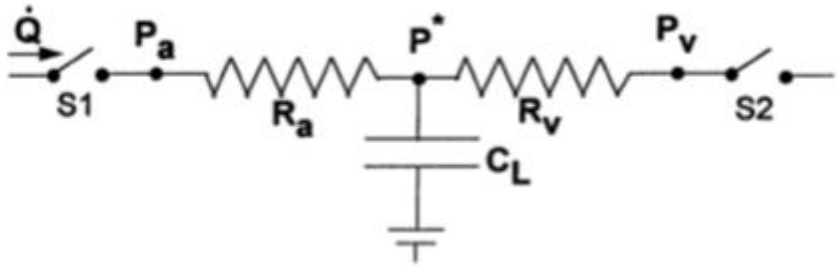


Figure 3.10: Electrical analog model representation of the pulmonary vasculature (Haworth S. T., 1996).

Here, P_a is the arterial pressure and P* is an estimate of the mean capillary pressure. The arterial and venous resistances are represented by R_a and R_v. At steady state, both switches S1 and S2 are closed. Following venous occlusion, blood continues to flow into the lung via the pulmonary artery; the pressure measured at the pulmonary vein shows an abrupt rise. Thereafter the arterial and venous pressures continue to rise slowly in a parallel manner (Haworth S. T., 1996). This behavior is explained by the model in **Figure 3.10** wherein the abrupt rise in P_v to P* would occur on opening S2. The value of P* can be obtained from the experimental data by extrapolation of the linear portion of the venous pressure curve back to the time of occlusion. This value is a useful estimate of pulmonary capillary pressure (Haworth S. T., 1996). The vascular compliance, $C_{L,}$ is

estimated from the slope of the venous pressure curve and blood flow. In double occlusion both switches, S1 and S2 are open, which means both the pulmonary artery and vein are occluded. After occlusion, the venous pressure abruptly increases and arterial pressure abruptly falls. As shown in the **Figure 3.10** they equilibrate at P*. When the inflow to the lobar pulmonary artery is occluded, P_a rapidly falls. This rapid fall is to P*, during the discharge of C_L (Haworth S. T., 1996).

3.5.3 Rarefaction

Reduction in the total number of blood vessels in a vascular bed (rarefaction) will increase vascular resistance by reducing the number of parallel pathways (Hopkins & McLoughlin, 2002). Rarefaction is proposed to be one of the factors leading to the increase in the pulmonary arterial pressure in the case of chronic hypoxia (Hopkins & McLoughlin, 2002). We have incorporated a simple model of rarefaction in order to study its effect on pulmonary arterial pressure by removing a specified fraction of arteries of a specific diameter. The user provides the diameter below which the vessels are to be removed and a value representing the fraction of the vessels in that size range to be removed. (Example: 50% of vessels with diameter under 100 microns are eliminated.)

3.5.4 Cardiac Output

In the dynamic model, pulsatile cardiac output is modeled by approximating the time dependent blood flow from the right ventricle of the heart as either a rectified sine wave or a sinusoidal wave. For the rectified sine wave, the systolic phase of blood flow is modeled as

$$Q = \overline{Q}\pi \sin\left(\frac{\pi\, fh\, t}{s/d}\right) \quad 3.60$$

whereas the sinusoidal cardiac output is modeled as

$$Q = \overline{Q}(1 + \sin(2\pi\, fh\, t)) \quad 3.61$$

where $\overline{Q}$ is the mean blood flow rate, *fh* is the frequency of the heartbeat, *s/d* is the systole to diastole blood flow ratio, and *t* is the time duration of the blood flow (Haworth S. T., 1996). In the rectified flow model, during the diastolic phase the right ventricular filling into the pulmonary artery is zero (Haworth S. T., 1996). Typical values for a rat are $\overline{Q}$ = 0.5 ml/sec, *fh* = 7 beats/second and *s/d* = 0.9 (Morell & Hughes, 2001).

3.6 Model Solution

3.6.1 Steady state solution

The steady state model solution is determined via an iterative process where inlet and outlet pressures, vascular resistance, vascular compliance, vascular inertance and vascular volumes in 18 orders of arteries, the capillary sheet and 19 orders of veins are calculated. The steady state pressure distribution and optimization process can be explained using the electrical analog circuit representing an artery, capillary and vein as shown in **Figure 3.11**.

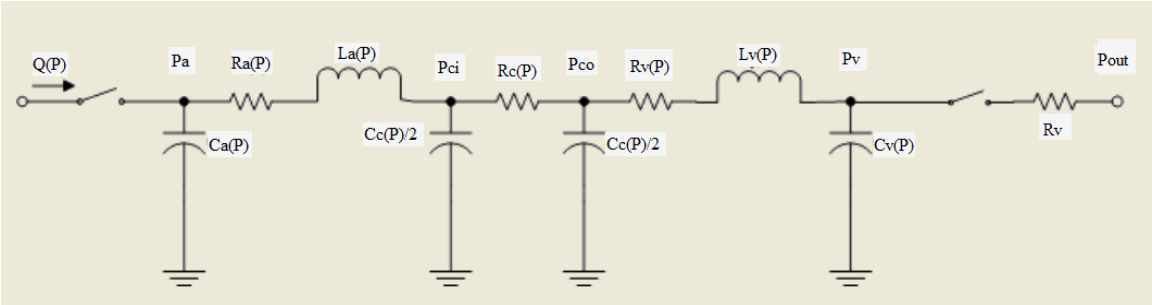


Figure 3.11: The electrical analog model representing arterial, capillary and venous components.

In **Figure 3.11**, Ra(P) and Rv(P) represent hemodynamic resistance of the artery and vein. La(P) and Lv(P) is the artery and vein inertances. Ca(P) and Cv(P) represent the arterial and venous compliance. Rc(P) and Cc(P) represent the capillary resistance and compliance. P_{ci} and P_{co} represent the inlet and outlet capillary pressures. P_{out} and R_v represent the output venous pressure and resistance.

To calculate the steady state pressure distribution within the model, steady state parameter values for mean flow (Q), venous pressure (P_{out}), alveolar pressure (PA) and pleural pressure (Ppl) are selected.

1. Since P_{out} and P_v are given, we can calculate exactly the value for P_v based on Ohm's Law:

$$P_v = QR_v + P_{out} \tag{3.62}$$

2. The capacitance (C_v) is calculated using the following empirical relationship (Haworth S. T., 1996):

$$C_v = \frac{\pi\alpha\ell\ ND\left(D_{MAX} - D\right)}{2(\gamma\ -1)} \tag{3.63}$$

were γ is the ratio of diameter at large transmural pressure (P_{tm}) to diameter at zero P_{tm}, α the distensibility coefficient, ℓ the length of the vessel, and D_{MAX} is the diameter at large P_{tm}.

3. The circuit element coefficients are found through optimizing the pressure of the order while maintaining constitutive relationships of diameter, length, and viscosity. The initial guess for P_{co} is then found by first assuming the value of P_v for P_{co}. Then, an average pressure for that segment is found as the average between the input and output pressures. Diameter $D(P_{co})$ and subsequently resistance, R_v are then found for the segment using the equations below:

$$D(P_{co})_v = D(0)_v\left[\gamma_v - (\gamma_v - 1)e^{\frac{-\alpha P_{co}}{\gamma_v - 1}}\right] \quad 3.64$$

$$R_v = \frac{128\mu\ell^2}{\pi N D_v^4} \quad 3.65$$

where γ is the ratio of maximum diameter at large P_{tm} to unstressed diameter, μ is viscosity, ℓ is the length of the segment, and α is the distensibility coefficient.

4. The new nodal pressure P_{co} is calculated using Ohm's Law, and this process is repeated until the change in P_{co} is below the desired tolerance. This iterative calculation continues for this order until the change with each iteration was less than 10^{-6} cmH_20.

$$P_{co} = QR_v + P_v \quad 3.66$$

5. Once P_{co} is estimated, inertance (L_v) is calculated from (Haworth S. T., 1996)

$$L_v = \frac{4\rho\ell_v}{\pi N D_v^2} \quad 3.67$$

where ρ is the fluid density.

6. Similarly, evaluation of the circuit components of the capillary bed (sheet) begin with the assumption of $P_{ci} = P_{co}$. The thickness and resistance are calculated for the new segment pressure (average of nodal pressures) as shown below. Again, using Ohm's law, the input pressure P_{ci} is recalculated and this process is repeated until the change in P_{ci} between iterations is below the desired tolerance.

$$h(P_{tm}, P_{tp}) = [h(0,0) - h(0,\infty)]e^{k_1 P_{tp}} + h(0,\infty)\left[\gamma - (\gamma - 1)e^{\frac{-\alpha P_{tm}}{(\gamma-1)}}\right] \quad 3.68$$

$$R_c = \frac{12\mu f \ell^2}{Ah^3 VSTR} \quad 3.69$$

$$P_{ci} = QR_c + P_{co} \quad 3.70$$

7. The compliance coefficient for the capillary sheet is given by (Haworth S. T., 1996):

$$C_c = \frac{VSTR\ A\alpha_c(h_{MAX} - h)}{2(\gamma_c - 1)} \quad 3.71$$

where h_{MAX} is the maximum capillary sheet thickness.

8. Steps 1 to 5 are repeated for each arterial order, starting with an initial guess of $P_{ci} = P_a$, recalculating the constitutive relations, and computing P_a based on Ohm's Law with a pressure drop of RQ.

3.6.2 Dynamic model solution

In the dynamic model, the vessels respond differently due to changes in diameter resulting from the input flow (pulsatile cardiac output) and ventilation and transient changes in blood flow and pressure. The steady state model solution is used as the basis for the dynamic model solution, but with the addition of the inertia of the blood and

nonlinear viscoelastic vessel properties. Determination of the coefficients in the dynamic model can be done using the constant or variable *RLC* model.

In the constant *RLC* version, the coefficients (*R*, *C* and *L*) are established at time *t* = 0. They remain constant independent of the functional form of blood flow, P_{tp} and the resulting volume and pressure changes.

The variable *RLC* model uses circuit elements with coefficients that are a function of pressure. Thus, in the variable *RLC* version, the coefficients of the differential equations are updated continuously, since they depend on vascular volume and pressure. As part of the numerical integration at each calculated pressure step, conservation of volume is enforced within each order. It is also necessary to update the circuit element coefficients, or differential equation coefficients, at every time step. Hence, the following steps are implemented.

1. The new volume is calculated using the definition of compliance:

$$C = \frac{\Delta V}{\Delta P} \qquad V_j = V_{j-1} + C\left(P_j - P_{j-1}\right) \qquad 3.72$$

where, the subscript *j* indexes the time values of volume for a given order.

2. The new diameter is calculated based on the new volume using the cylindrical volume-diameter relationship (Haworth S. T., 1996):

$$D_j = \sqrt{\frac{4V_j}{\pi \ell_j N_j}}. \qquad 3.73$$

3. The circuit element coefficients (*R*, *C* and *L*) are calculated based on the new diameter:

$$R_j = \frac{128\mu_j \ell_j}{\pi N_j D_j^4} \qquad 3.74$$

$$C_j = \frac{\pi \ell_j N_j \alpha_j D_j \gamma_j \left(D(0)_j - D_j\right)}{2\left(\gamma_j - 1\right)} \quad 3.75$$

$$L_j = \frac{4\rho \ell_j}{\pi N_j D_j^2} \quad 3.76$$

where the subscript *j* indicates the order of the vessel tree.

Viscoelasticity model

Krishnan, et al. (Krishnan, Linehan, Rickaby, & Dawson, 1986) found that the purely elastic behavior assumption for the vessel wall was inappropriate during dynamic events. For this, they substituted the St. Venant element as shown in **Figure 3.12** to include the effects of viscoelasticity. The St. Venant element behaves similarly to a parallel spring/dashpot system, where upon a change in pressure, the C_b element will immediate react in an elastic manner, and as the pressure drop increases across R_w, the viscoelastic capacitor C_w will then either charge or discharge accordingly over some finite period of time. In larger pulmonary vessels this effect is minimal.

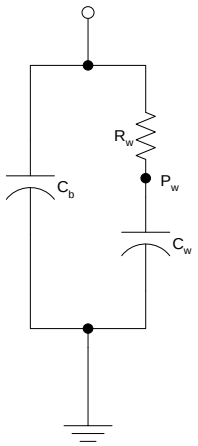


Figure 3.12: Analog equivalent of St. Venant viscoelastic element.

For the element above, the following three linear, first-order differential equations can be written:

$$P_j - P_{j+1} = L_{j+1}\frac{dQ_{j+1}}{dt} + R_{j+1}Q_{j+1} \tag{3.77}$$

$$\frac{P_j - P_{w_j}}{R_w} = C_{wj}\frac{dP_{wj}}{dt} \tag{3.78}$$

$$Q_j = Q_{j+1} + C_{bj}\frac{dP_j}{dt} + \left(\frac{P_j - P_{w_j}}{R_j}\right) \tag{3.79}$$

which can be rewritten into the following form below

$$\frac{dQ_{j+1}}{dt} = \frac{\left(P_j - P_{j+1}\right)}{L_{j+1}} - \frac{R_{j+1}}{L_{j+1}}Q_{j+1} \tag{3.80}$$

$$\frac{dP_{w_j}}{dt} = \frac{P_j - P_{w_j}}{R_w C_{b_j}} \tag{3.81}$$

$$\frac{dP_k}{dt} = \frac{Q_j - Q_{j+1}}{C_{d_j}} - \left(\frac{P_j - P_{w_j}}{R_w C_{d_j}}\right) \tag{3.82}$$

where the subscript j is indicates the order of the vessel tree and w_j indicates the viscoelasticity parameter at a particular vessel order. The differential equations for pressure in the capillary sheet are given by

$$\frac{dP_j}{dt} = \frac{2}{C_n}\left(\frac{P_{j-1} - P_j}{R_{n-1}} - \frac{P_j - P_{j+1}}{R_n}\right) \tag{3.83}$$

$$\frac{dP_{w_j}}{dt} = \frac{1}{C_{w_j}}\left(\frac{P_j - P_{w_j}}{R_w}\right) \tag{3.84}$$

where the subscript j represents the number of capillary sheets.

This viscoelastic element, then replaces the capacitor elements throughout the analog model of small pulmonary vessels as shown in **Figure 3.13**. The consolidated electrical analog lung model for rat is shown in the **Figure 3.14**.

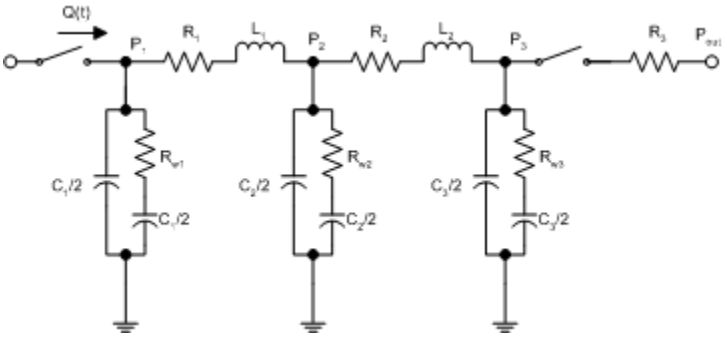


Figure 3.13: Circuit for Viscous *RLC* model.

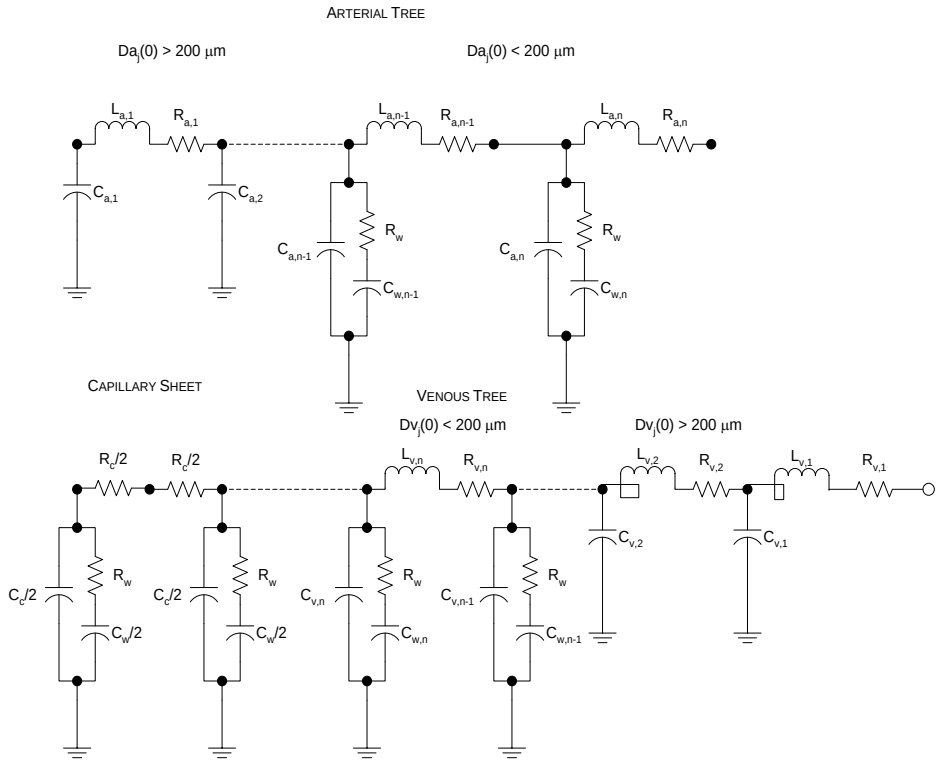


Figure 3.14: The electrical analog of the rat lung model.

For the dynamic model, there are 3 differential equations for each arterial and venous order and 2 differential equations for each capillary order. Thus for this rat model with 18 orders of arteries, 19 orders of veins and the capillary sheet, the model consists of

a system of 113 first order differential equations. These equations are solved using Gear's method (Haworth, Linehan, Bronikowski, & Dawson, 1991) for a system of stiff differential equations (ode15s in MATLAB, Inc). The resulting solution gives the pressure distribution within arteries, veins and capillaries and the flow distribution within arterial and venous segments based on a given cardiac output and other input variables.

4. SOFTWARE DESIGN

4.1 System overview

Haworth developed preliminary software that implements the pulmonary hemodynamics model presented above using data from dogs (Haworth S. T., 1996). In the work presented below, we present an overview of the model software developed here. The software is implemented in MATLAB 7.14 (R2012a) and designed to be user-friendly. It includes a graphical user interface for choosing parameter, simulation, and numerical options for performing either steady state or dynamic simulations. The output results can be visualized by the user with an interactive interface, or exported to perform custom analysis. The model is used to calculate the pressure and flow distribution throughout the lung for a specified period of time, based upon the selected input parameters, such as mean flow, venous pressure, and cardiac and breathing behavior.

The software, referred to as the "System", consists of three subsystems: *Input, Model* and *Result.* Subsystem *Input* is a user interface module that stores and loads the values of the inputs required to run the model. Subsystem *Model* contains the differential equations that are solved for the pressure and flow distributions within the lung based on the selected inputs. Subsystem *Result* organizes the output generated from the core module and provides display options. Each of these subsystems is explained below.

4.2 Subsystem Input

The responsibility of subsystem *Input is* to pass the model inputs selected by the user to the subsystem *Model* for running of the simulation. The user-selected inputs, model options and the functional model options are stored in an input parameter structure (‘gsim_Params’) and are fed into the main program or the model.

The Subsystem *Input* consists of four modules: Simulation User Interface (UI), Options User Interface, Parameters and Constants, and Geometry as shown in **Figure 4.1**. Simulation UI is for model inputs by the user. The Options UI is used to enter the functional model options and additional model parameters. The geometry module stores the morphometry data file which is selected by the user. The constants used in the model are stored in a structure called ‘constants’. These selections are stored in an input parameter structure called ‘gSim_Params’. This input structure is passed to the Subsystem *Model* to run the simulation based on the selection.

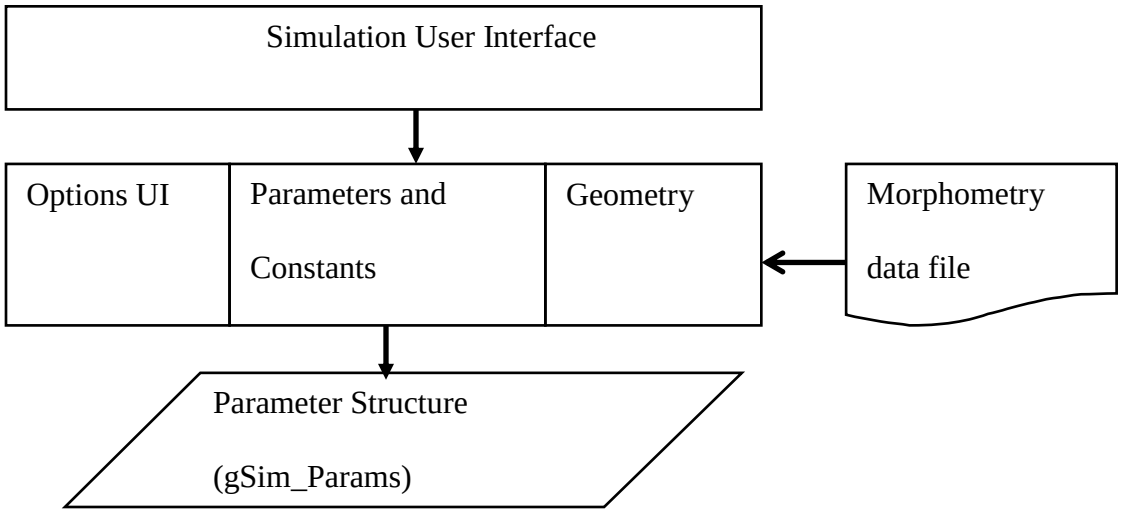


Figure 4.1: Flow chart representing the data flow between the different modules of Subsystem *Input*.

Each module of the subsystem *Input* is explained in detail below.

4.2.1 Simulation User Interface (Simulation UI)

The Simulation User Interface is the main user interface for interacting with the system. It provides place holders for accepting input, running the simulation and viewing the results. It also has options to add or delete multiple simulations and run at the same time. It provides user interfaces for invoking the following functionalities.

1) Accept Input parameters and Model Options: Model parameters and functional model options can be entered manually or loaded from an existing MatLab file.
2) Run the Simulation: The main subsystem *Model* is called by clicking this button to run the simulation based on the parameters and options selected.
3) View Results: The result subsystem is called by clicking this button to view, save or export the model simulation results.
4) Add/Delete Multiple Simulations: This button can be used to run multiple simulations at the same time by copying existing parameters and options or by manual entry.

The screen shot of the Simulation User Interface is in **Figure 4.2**. The layout of Simulation UI consists of an Input and Result section, and buttons to run and exit the simulation. The sections of the Simulation UI are explained below:

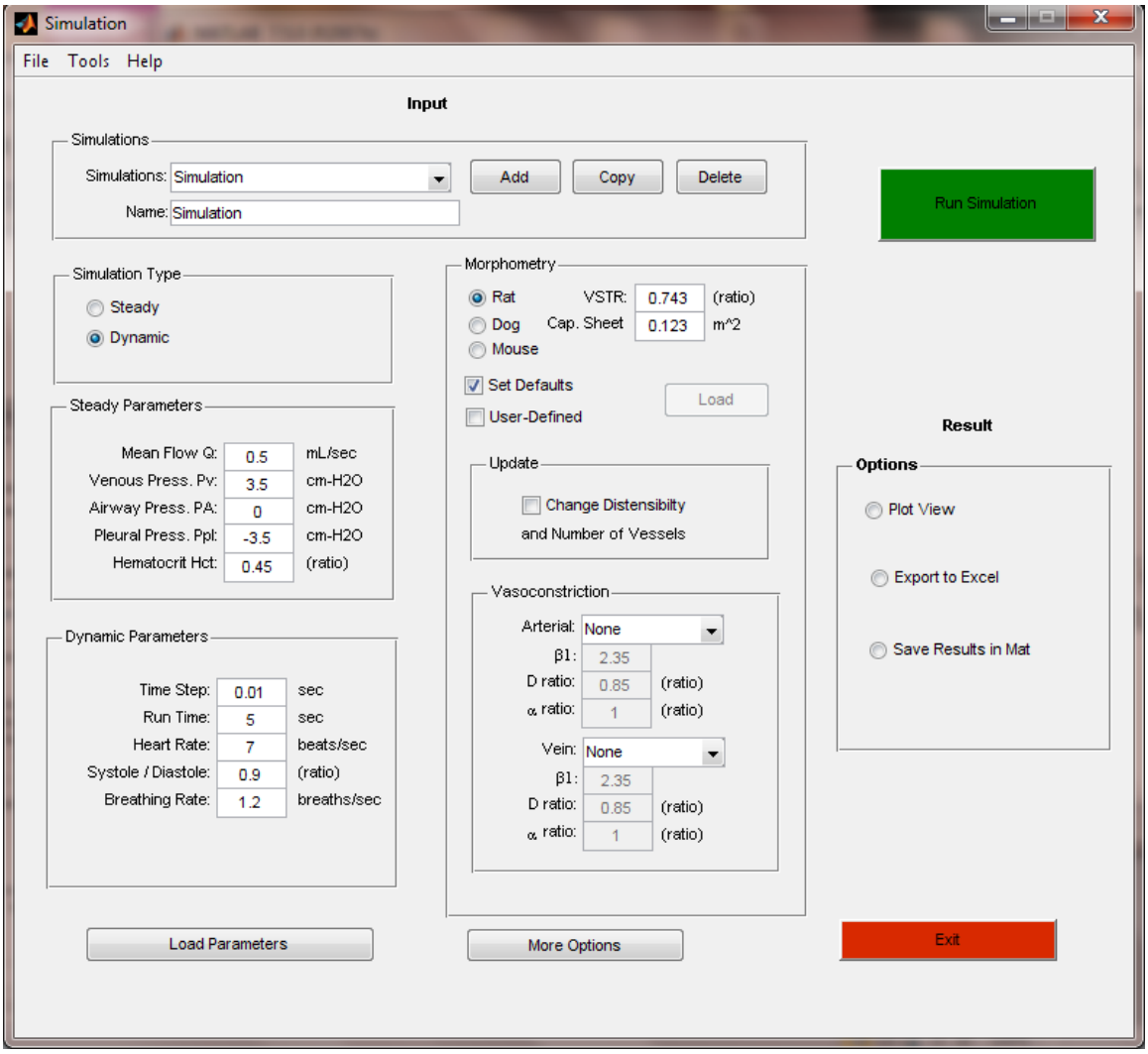


Figure 4.2: Screenshot of the Main Simulation User Interface.

Input Section:

*Simulations***:** This section allows the user to add names for the simulation and run multiple simulations. It consists of a combo box, edit box and three buttons. Multiple simulation options can be done using the Add, and Copy buttons as shown in **Figure 4.2**. A simulation can be deleted using the Delete button. When the user clicks Add, a new structure is created with the default existing parameters and options. The Copy button

creates a new structure with the parameters and options of the previous simulation selection. Also, the user can give specific names for each simulation under “Name”.

Simulation Type: radio buttons to select either steady state or dynamic simulation options.

Steady State Parameters: edit boxes to accept parameters values to run the steady state model.

Dynamic Parameters: edit boxes to accept parameter values for running the dynamic model.

Load Parameters: enter all model parameters and options in the Simulation UI from an existing matlab file. The different steady state and dynamic parameters are described in section III, Parameters and Constants.

Morphometry: radio button to load rat morphometric data. The user can choose the default selection of the rat file or manually load other data using the load button. There are also options to load mouse, or dog morphometric files.

Update: used to modify the vessel morphometry characteristics (distensibility and number of vessels) in a selected rat file using a GUI explained in section IV, Geometry.

Vasoconstriction: option to change the diameter and length of arteries and veins of different diameters based on the selected variables.

More Options: invokes a GUI for selecting different functional models and additional dynamic parameters for running the simulation as described in section II, Options User Interface.

Run Simulation: selected parameters and options are stored in the structure ‘gSim_Params’ and passed to the Subsystem Model to use to run the simulation.

Results section:

The results of the model simulation are stored in a structure called 'gSim_Results'. The Result section consists of three options.

1) Plot View: generates a GUI to interactively plot the results of the simulation.
2) Export to excel: organizes results and writes to an excel spreadsheet.
3) Save results: organizes results and parameters and saves to a matlab file. The parameters of this saved file can be reloaded to the Simulation GUI using the load button for subsequent simulation testing.

Exit: exits the Simulation UI.

4.2.2 Options User Interface

This module provides the user interface for selecting different functional models and entering additional dynamic parameters for running the simulation. **Figure 4.3** shows that the UI for the Options GUI is divided into Additional parameters and Functional models.

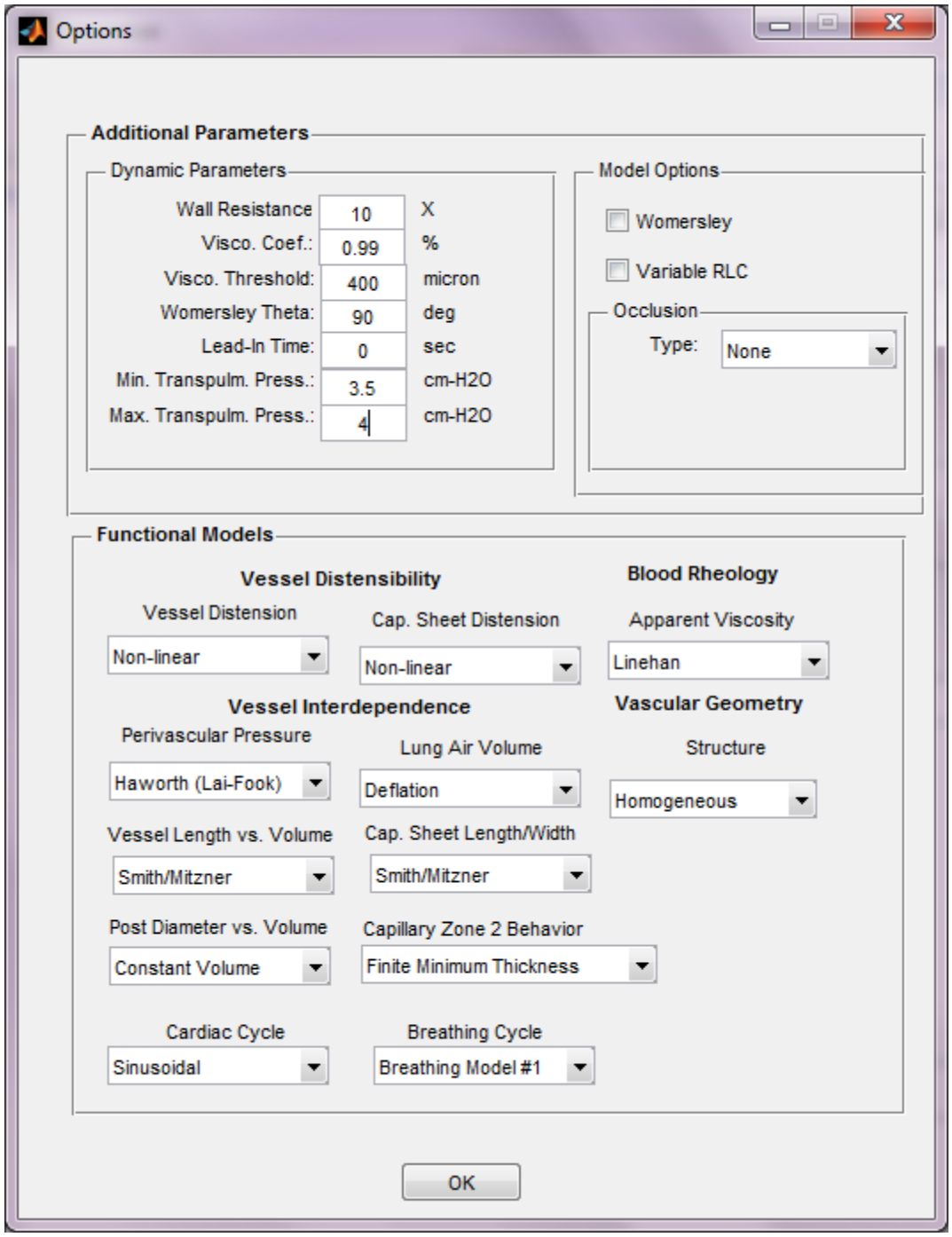


Figure 4.3: Screen shot of the Options User Interface.

Additional Parameters: This is divided into two sections - Dynamic Parameters and Model Options. The Dynamic parameter section includes edit boxes to enter additional

dynamic input listed in **Table 4.1**, to run the dynamic simulation. The model option section gives the user three dynamic simulation options:

1) Womersley: uses the Womersley equations to calculate resistance.
2) Variable *RLC*: updates the coefficients during simulation.
3) Occlusion options: select from i) no occlusion, ii) arterial occlusion, iii) venous occlusion, iv) double occlusion. The duration of the occlusion is entered in the edit box.

Functional Models: This section in the Options UI is divided into four sections: i) vessel distensibility, ii) vessel interdependence, iii) blood rheology, and iv) vascular geometry. The options for each section are provided as a list in the drop down box. The list of functional models in each section is explained in the Subsystem - *Model*, Section - Functional Models.

4.2.3 Parameters and Constants

This module handles all the input (steady and dynamic) parameters and constants used in the model. The parameter values are stored in the structure 'Model Parameters' and the constants are stored in the function 'constants' and then stored as default values in the 'gsim_Params' structure when running the Simulation UI. The list of constants used in the model simulation is given in Appendix 1(b). The Module parameter consists of two subsystems: Steady state Parameters (SS) and Dynamic Parameters. **Table 4.1** shows the list of SS and Dynamic input parameters used in the model simulation.

Steady State Parameters(Units)	Dynamic Parameters(Units)
Mean Flow *Q* (ml/sec)	Time step (seconds)
Venous Pressure *Pv* (cm-H_20)	Run Time (seconds)
Airway Pressure *PA* (cm-H_20)	Heart rate (beats/sec)
Pleural Pressure *Ppl* (cm-H_20)	Systolic/Diastolic ratio (Ratio)
Hematocrit (Ratio)	Breathing Rate (breaths/sec)
	Wall resistance (multiplier for Rw element)
	Viscosity coefficient (Ratio)
	Viscoelasticity threshold (microns) Diameter cutoff to apply viscoelasticity.
	Womersley theta (degrees)
	Min Transpulmonary pressure (cm-H_20)
	Max Transpulmonary pressure (cm-H_20)
	Lead in time (seconds)

Table 4.1: Steady state and dynamic parameter inputs.

4.2.4 Geometry

This module loads the morphometry file based on the user selection in the Simulation UI and stores it in the 'gSim_Params' structure. Appendix 1(a) show the input rat morphometry file for the model. For hypothesis testing, the arterial and venous distensibility and the number of arteries can be changed using the Update button in the Simulation UI. This button invokes a new GUI called 'Morphometry_File_Change' as shown in **Figure 4.4**.

1) **Distensibility:** Once the file is loaded, current values of arterial and venous distensibility are shown and can be edited. The update button will modify the morphometric file used as input to run the simulation.

2) **Arterial Rarefaction:** This module is called by clicking the Rarefaction- Arteries checkbox. The user can enter the diameter range and the percentage reduction. For example: entering 50 and 100 in the percentage and diameter edit boxes mean eliminate 50% of the vessels with diameter under 100 microns. The update button will update the morphometric file used as input.

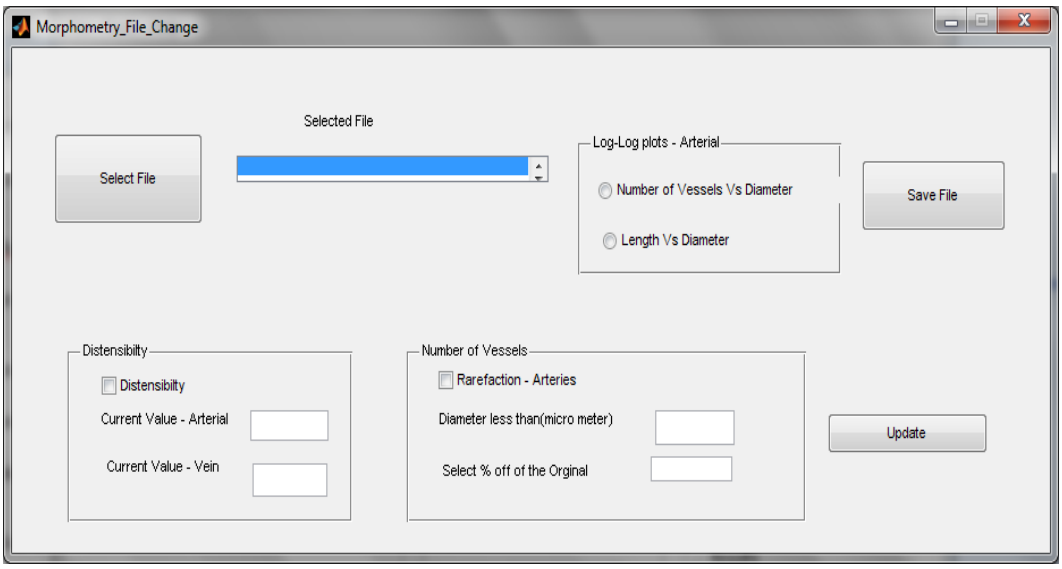


Figure 4.4: GUI for changing the distensibility and number of arteries in the input morphometric file.

4.3 Subsystem – Model

The subsystem *Model* consists of the steady state, dynamic, and functional models. Input is from the 'gSim_Params' structure from the Subsystem *Input*. The steady state model is run first using values from the input parameter structure. These results and the input dynamic parameters are then used to run the dynamic model. The functional model functions are common to both the steady state and dynamic model.

Figure 4.5 shows the sequence and data flow across the modules of subsystem *Model*. Each of these modules of Subsystem *Model* is explained below.

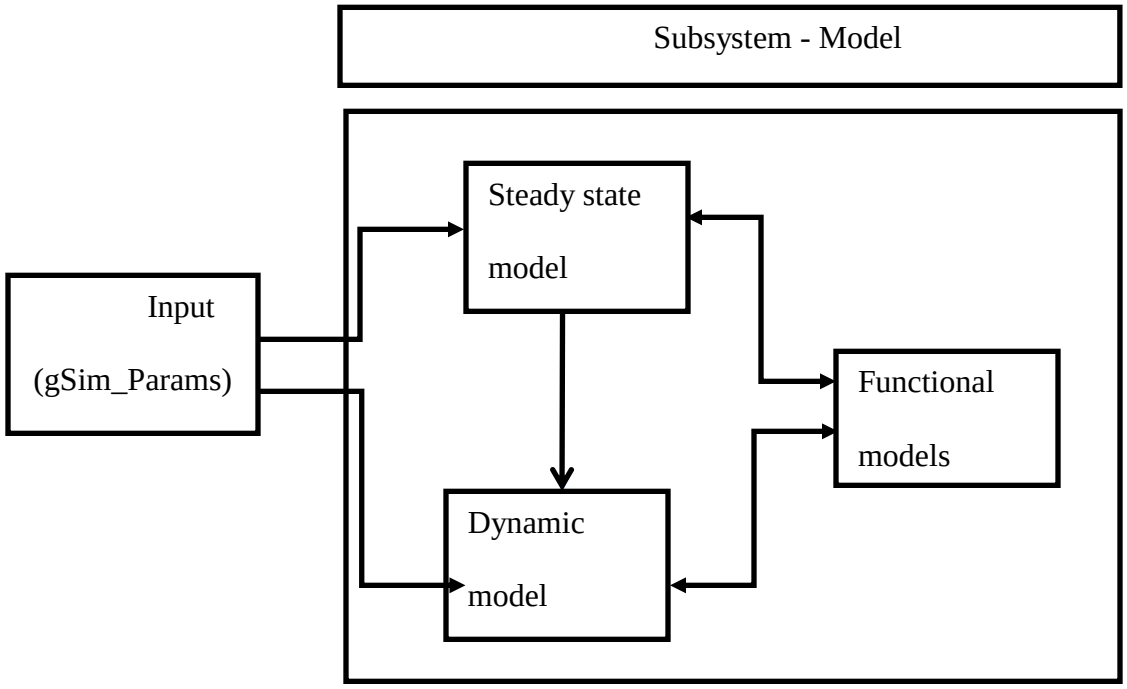


Figure 4.5: Flow chart representing data flow between the different modules of Subsystem *Model*.

4.3.1 Steady State Model

The steady state model calculates inlet and outlet pressures, vascular resistance, vascular compliance, vascular inertance and vascular volume at each of the 18 orders of arteries, the capillary sheet and 19 orders of veins for the rat model by solving the differential equations presented in Section 3.6.1.

The main function to calculate the steady state pressure distribution of arteries and veins is FUN_ART_VEIN and for capillaries is FUN_CAP as shown in **Table 4.2**.

Function name	Syntax	Description
FUN_ART_VEIN	[Pressure] = FUN_ART_VEIN(gSim_Params)	Solves the Poiseuille pressure equations for arterial and venous vessel trees.
FUN_CAP	function [Pressure_Cap] = FUN_CAP(gSim_params)	Calculates capillary bed pressure.

Table 4.2: Functions for calculating the steady state pressure distributions with the syntax and description.

4.3.2 Dynamic Model

Inputs to the dynamic model include the steady state model results as well as the cardiac output and nonlinear viscoelastic properties. Our rat model is represented by a system of 113 first order differential equations, which is solved using Gear's method for stiff (ode15s in MATLAB, Inc). The resulting solution output consists of the arterial, venous and capillary pressure distribution as well as blood flow within each arterial and venous segment.

4.3.3 Functional Models

The functional model functions common to both the steady state and dynamic model are shown in **Table 4.3**. The equations for calculating these functions are given in Chapter 3.

Functional models	Syntax	Calculation
Compliance – Arteries/Veins	[C] = F_Compliance_ArtVen(gSim_Params)	Compliance of vessels. Output: variable C
Compliance - Capillaries	[C] = F_Compliance_Cap(gSim_Params)	Compliance of capillaries Output: variable C
Inertance -	[L_Induct] =	Inertance of vessels

Arteries/Veins	F_Inductance_ArtVen(gSim_Params)	Output: L_Induct
Resistance – Arteries/Veins	[R] = F_Resistance_ArtVen(gSim_Params)	Resistance of vessels. Output: variable R
Resistance - Capillaries	[R] = F_Resistance_Cap(gSim_Params)	Resistance of capillary sheet. Output: variable R.
Air Volume	[V_Ptp, V_Ptp_ref] = F_Model_AirVolume(gSim_Params)	Lung volume (fn of transpulmonary pressure) Output: V_Ptp and V_Ptp_ref (reference volume)
Capillary sheet height	[hc] = F_Model_CapSheet_Height(gSim_Params)	Capillary sheet height (hc) (fn of transpulmonary and transmural pressure)
Capillary sheet length and width	[Lc, Wc] = F_Model_CapSheet_Length_Width(gSim_Params)	Capillary sheet length and width (fn of volume)
Capillary sheet post diameter	[epsilonc] = F_Model_CapSheet_PostDiam(gSim_Params)	Capillary sheet post diameter (fn of volume)
Vessel Diameter Arteries/Veins	[D] = F_Model_Diam(gSim_Params)	Vessel diameter D (fn of transmural pressure)
Cardiac cycle	[Q] = F_Model_CardiacCycle(gSim_Params)	Cardiac cycle model
Capillary model friction factor	[f] = F_Model_FrictionFactor(gSim_Params)	Geometric friction factor (fn of capillary dimensions)
Vessel length	[L] = F_Model_Length(gSim_Params)	Vessel length (fn of lung volume)
Perivascular pressure	[Px] = F_Model_PxHat(gSim_Params)	Perivascular pressure
Blood viscosity	[mu_a] = F_Model_Viscosity(gSim_Params)	Apparent viscosity (fn of vessel diameter)
Volume Arteries/veins	[V] = F_Volume_ArtVen(gSim_Params)	Vessel volume (fn of diameter and length)
Volume Capillaries	[V] = F_Volume_Cap(gSim_Params)	Capillary sheet volume (fn of sheet thickness, area and vascular space to tissue ratio (VSTR))

Table 4.3: List of functional models with their syntax and description.

4.4 Subsystem – Results

The results of the model simulation are handled by the Subsystem - *Results*. It has three modules.

1) View Results: GUI to interactively view the results of the steady state and dynamic model simulations.
2) Export to Excel: writes the parameters and results of the model simulation to an excel file.
3) Save Results: organizes and saves parameters and results of the model simulation to a matlab file.

The results generated from the model simulation are organized in a structure called 'gsim_Results'. For multiple simulations, multiple structures are created inside the 'gSim_Results' to store the results. The parameters used for the simulation are stored in 'gSim_Parameters'. **Table 4.4** shows the results generated by the model simulation. The three different modules in the result section are explained below.

Steady State Simulation Results	Dynamic Simulation Results
Nodal Pressure (cm-H_20)	Nodal Pressure (cm-H_20)
Mean Pressure (cm-H_20)	Mean Pressure (cm-H_20)
	Flow (ml/sec)
Resistance (cm-H_20sec/ml)	Resistance (cm-H_20sec/ml)
Inertance (cm-$H_20$$ml^{-1}s^2$)	Inertance (cm-$H_20$$ml^{-1}s^2$)
Compliance (ml/cm-H_20)	Compliance (ml/cm-H_20)

Table 4.4: Results generated for the arteries, capillaries and veins by the model simulation.

4.4.1 Module – View Results

This module provides a GUI to view the simulation results. It is called when the user clicks the 'Plot View' radio button in the Simulation UI. **Figure 4.6** shows a screen

shot of the steady state and dynamic results, in the Result view GUI. The steady state and dynamic model results are displayed in the top left and bottom left panels, respectively.

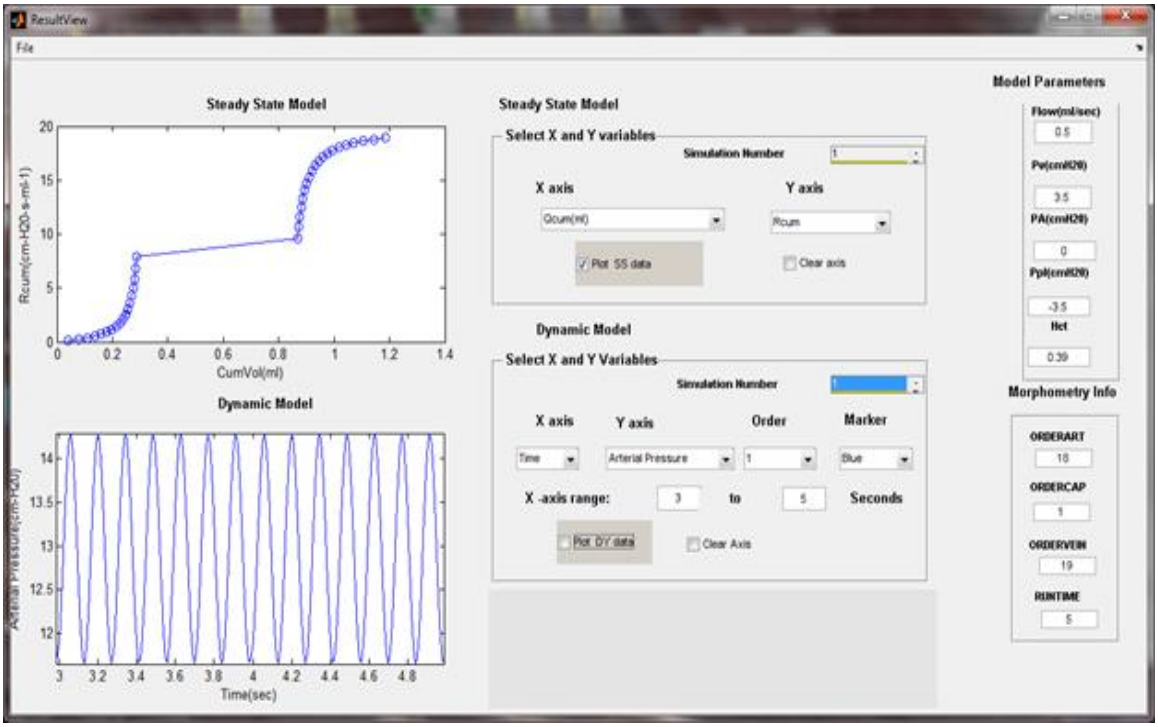


Figure 4.6: Screenshot of the GUI for entering result parameters and plotting simulation results.

The right side of the result view GUI has four sections: Model Parameters, Morphometry Info, Steady Model and Dynamic Model.

Model Parameters: Steady state model parameters including mean flow, venous pressure, alveolar pressure, pleural pressure and hematocrit ratio.

Morphometry Info: Includes the number of orders of arteries, veins and capillaries, and displays run time for the dynamic simulation.

Steady State Model: Option to select steady state simulation result variables for plotting in the result top panel. The x-axis variable options for the steady state simulation plots are i) Order (Number), ii) Cumulative volume (Vcum), or iii) Mean flow (Flow).

The y-axis options are i) Nodal pressure (Pnode), ii) Mean pressure (Pmid), iii) Cumulative resistance (Rcum), iv) Cumulative Inertance (Lcum), v) Cumulative Compliance (Ccum), or vi) Pulmonary arterial pressure (*Pa*).

Simulation Number: The indexes of the number of simulations run.

After selecting the Simulation number and the x- and y- axis variables, the user checks the plot SS data check box. The results of the selection are plotted in the top panel as shown in the **Figure 4.6**. In this case, the figure shows cumulative resistance versus cumulative volume. The axis can be cleared using the clear axis check box and can be re-plotted again with different user-selected results.

Dynamic Model: This section is used to select the dynamic simulation result variables, for plotting in the result in the bottom panel. The following are the options for the dynamic model plots.

Simulation number: The index of the number of simulations run. It consists of a list box to select a result structure from a list of multiple simulation result structures.

Dynamic X axis variable: The x-axis variable is Time (seconds).

Dynamic Y axis variables: The y-axis variables can be selected from Pulmonary Arterial pressure, Venous pressure, Capillary Pressure, Arterial Flow, Venous Flow or Cardiac Output.

Order: Select plots for a particular vessel order. The information on the arterial, venous and capillary orders are shown in the morphometry info section.

Marker: Plots can be given different symbol colors (red, blue, green, cyan, magenta, yellow and black) using the marker drop down menu icon.

X-axis Range: The x-axis range of the dynamic simulation can be entered in the edit boxes. For example, the bottom panel displays the plot of pulmonary arterial pressure (Order 1) versus time for the 3 to 5 second interval.

After selecting the simulation number, x- and y- axis variables, order, marker and range, the user checks the 'plot DY data' check box. The results are plotted in the Dynamic Model plot axis (bottom panel) as shown in the **Figure 4.6**. The axis can be cleared using the clear axis check box and re-plotted with different selections.

The function used to perform the operation of view result for this operation is [status] = View_Results(gSim_Results). The input to this function is the 'gSim_Results' structure and the output is a status variable. It returns either 1 or -1 depending on whether the data is transferred to the View_Results GUI or not.

4.4.2 Module – Export to excel

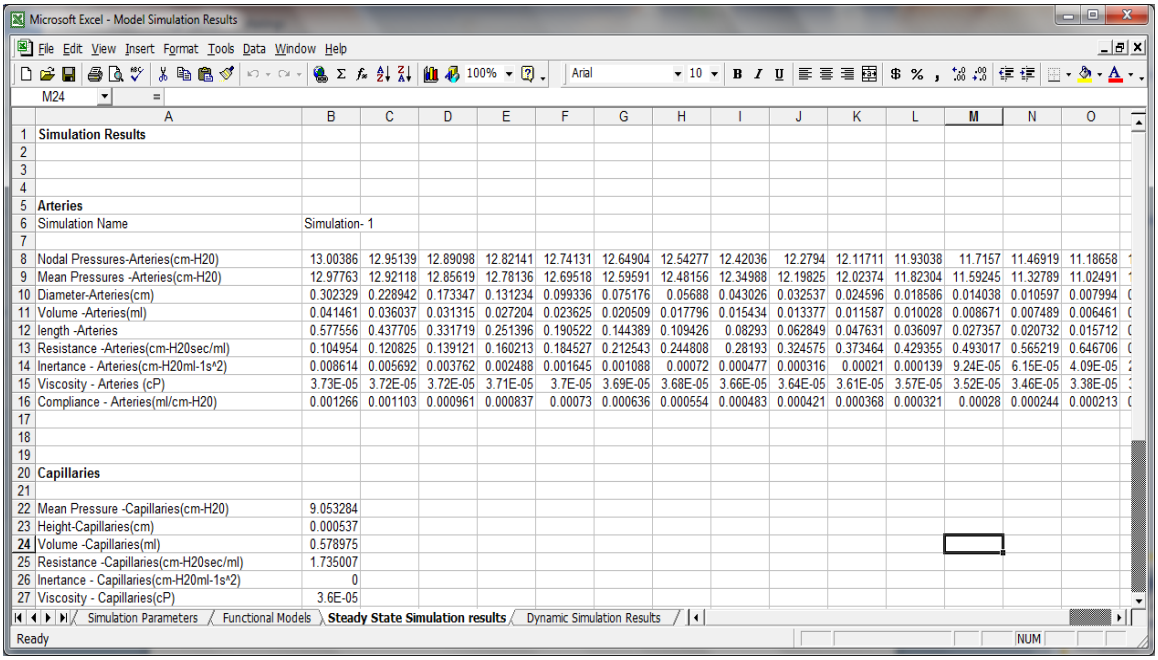
This module is called by clicking the 'Export to excel' radio button in the Result option section of the Simulation UI. The user is prompted to select an excel file to write the simulation results. Then four tabs are created in the excel sheet. **Figure 4.7:** shows a screenshot of the excel sheet where the four tabs are:

Simulation Parameters: steady state and dynamic parameters used for the model simulation.

Functional Models: options used for the simulation.

Steady State Simulation Results: steady-state results are organized under the headings Arteries, Capillaries and Veins with the corresponding Simulation name and number.

Dynamic Simulation Results: dynamic results and their values.



Microsoft Excel - Model Simulation Results

File Edit View Insert Format Tools Data Window Help

M24

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	**Simulation Results**														
2															
3															
4															
5	**Arteries**														
6	Simulation Name	Simulation- 1													
7															
8	Nodal Pressures-Arteries(cm-H20)	13.00386	12.95139	12.89098	12.82141	12.74131	12.64904	12.54277	12.42036	12.2794	12.11711	11.93038	11.7157	11.46919	11.18658
9	Mean Pressures -Arteries(cm-H20)	12.97763	12.92118	12.85619	12.78136	12.69518	12.59591	12.48156	12.34988	12.19825	12.02374	11.82304	11.59245	11.32789	11.02491
10	Diameter-Arteries(cm)	0.302329	0.228942	0.173347	0.131234	0.099336	0.075176	0.05688	0.043026	0.032537	0.024596	0.018586	0.014038	0.010597	0.007994
11	Volume -Arteries(ml)	0.041461	0.036037	0.031315	0.027204	0.023625	0.020509	0.017796	0.015434	0.013377	0.011587	0.010028	0.008671	0.007489	0.006461
12	length -Arteries	0.577556	0.437705	0.331719	0.251396	0.190522	0.144389	0.109426	0.08293	0.062849	0.047631	0.036097	0.027357	0.020732	0.015712
13	Resistance -Arteries(cm-H20sec/ml)	0.104954	0.120825	0.139121	0.160213	0.184527	0.212543	0.244808	0.28193	0.324575	0.373464	0.429355	0.493017	0.565219	0.646706
14	Inertance - Arteries(cm-H20ml-1s^2)	0.008614	0.005692	0.003762	0.002488	0.001645	0.001088	0.00072	0.000477	0.000316	0.00021	0.000139	9.24E-05	6.15E-05	4.09E-05
15	Viscosity - Arteries (cP)	3.73E-05	3.72E-05	3.72E-05	3.71E-05	3.7E-05	3.69E-05	3.68E-05	3.66E-05	3.64E-05	3.61E-05	3.57E-05	3.52E-05	3.46E-05	3.38E-05
16	Compliance - Arteries(ml/cm-H20)	0.001266	0.001103	0.000961	0.000837	0.00073	0.000636	0.000554	0.000483	0.000421	0.000368	0.000321	0.00028	0.000244	0.000213
17															
18															
19															
20	**Capillaries**														
21															
22	Mean Pressure -Capillaries(cm-H20)	9.053284													
23	Height-Capillaries(cm)	0.000537													
24	Volume -Capillaries(ml)	0.578975													
25	Resistance -Capillaries(cm-H20sec/ml)	1.735007													
26	Inertance - Capillaries(cm-H20ml-1s^2)	0													
27	Viscosity - Capillaries(cP)	3.6E-05													

Simulation Parameters | Functional Models | **Steady State Simulation results** | Dynamic Simulation Results

Ready NUM

Figure 4.7: Screenshot of the excel sheet with the simulation results. The four tabs Simulation Parameters, Functional Models, Steady State Simulation Results and Dynamic Simulation results are shown. The tab highlighted shows the steady state simulation results.

The function used for this operation is [status] = ExcelExport(gSim_Results). The input to the function is the ‘gSim_Results’ structure. The output is a status variable; either 1 or -1 based on whether the data was written successfully to the excel sheet or not.

4.4.3 Module – Save Results

This module is called when the user selects the ‘Save Results’ radio button in the ‘Result Option’ section of the Simulation UI. The results and parameters of the model simulation are saved as matlab (.mat) files. The parameter file can be reloaded using the’ Load’ button in order to rerun the simulation file. The function used for this operation is

[status] = Save_Results(gSim_Results). The input to this function is the gSim_Results structure. The output is a status variable; either 1 or -1 based on whether the data was saved successfully to the specified location.

5. MODEL TESTING AND VALIDATION

5.1 Model Testing

The model was tested by performing simulations of hemodynamic experiments for which the results have been previously measured or can be reliably predicted. These tests provide a comparison of the model responses with those of the lung itself. The input model parameters were obtained from the literature for typical control values of a 0.33-kg rat measured at functional residual capacity (*FRC*) (Sasaki, Yasuda, McCully, & LoCicero, 1997). The heart rate, breathing rate and systolic/diastolic ratio (additional inputs to the dynamic model) shown in **Table 5.1** are also in the normal range.

Input parameters (Units)	Model values
Mean Flow (ml/sec)	0.5
PA (cm-H_20)	0
Pv (cm-H_20)	3.5
Ppl (cm-H_20)	-3.5
Hct (ratio)	0.45
Heart rate (beats/sec)	7
Breathing rate (breaths/sec)	1.2
Systole/Diastole (ratio)	0.9

Table 5.1: Model input parameters values for a control rat.

Sections 5.1.1 to 5.1.4 describe the simulations performed to verify the reliability of this large scale rat model using the inputs in **Table 5.1**.

5.1.1 Rigid vs. Distensible model

We tested the rigid vessel model by setting the distensibility coefficient (α) to 0.0001 for both arteries and veins, and ran the model for flows ranging from 0 to 2 ml/sec with the model parameters given in **Table 5.1**. **Figure 5.1**[Top] shows the resulting arterial pressure (*Pa*) versus flow relationship. The results indicate strong linearity which shows that as the flow is doubled; the pressure is also doubled, typical of a rigid tube model (Linehan, Haworth, Nelin, Krenz, & Dawson, 1992). We then tested the distensible vessel model by setting the arterial and venous distensibility coefficients to reported experiment values for a control rat (α_A = 2.8%/ mmHg (Clarke, Baumgardt, & Molthen, 2010) and α_V =1.6 %/ mmHg (Molthen, Gordon, Krenz, & Clough, 2007)). **Figure 5.1**[Bottom] shows a nonlinear arterial pressure versus flow result. Note that as flow increases, pressure increases, but that the rate of increase is larger at low flows and then appears nearly constant for flows greater than ~0.8 ml/sec.

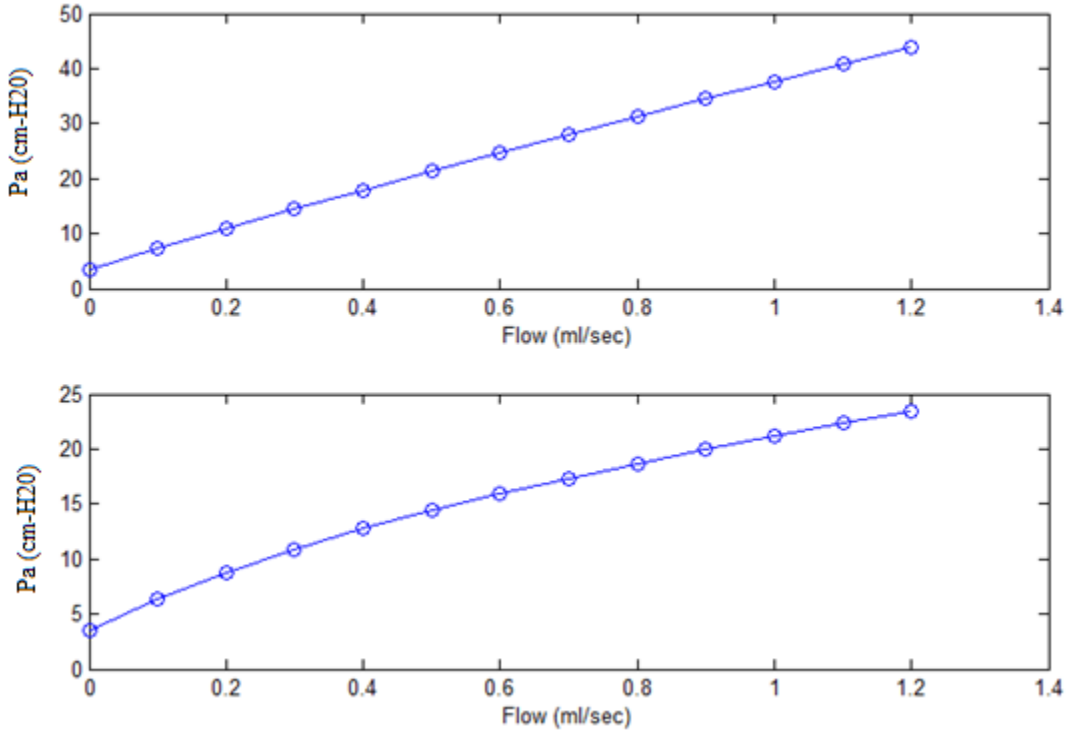


Figure 5.1: Pulmonary arterial pressure versus flow for the rigid vessel simulation (α_A = α_V = 0.0001) [Top], and for the distensible vessel simulation (α_A = 2.8%/mmHg; α_V = 1.6%/mmHg) [Bottom].

5.1.2 Conservation of flow:

The principle of flow conservation is assumed in this model. To test this assumption in the dynamic model, flow pulses with means of 0.5 and 1 ml/sec were generated (**Figure 5.2**) and used as input (cardiac output) to the lung model. The objective was to determine if the mean flow is the same at each order of arteries and veins.

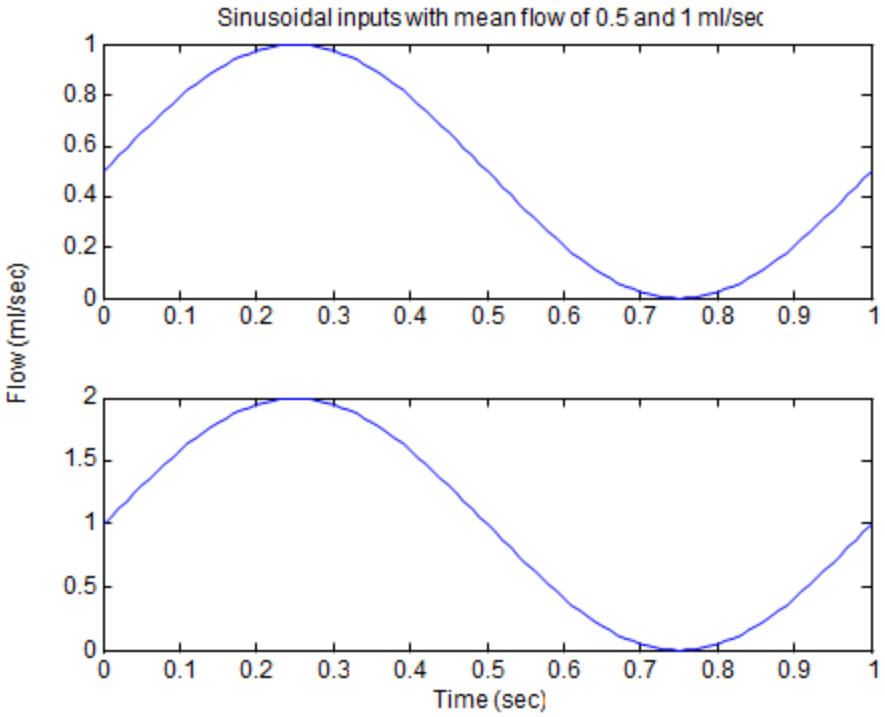
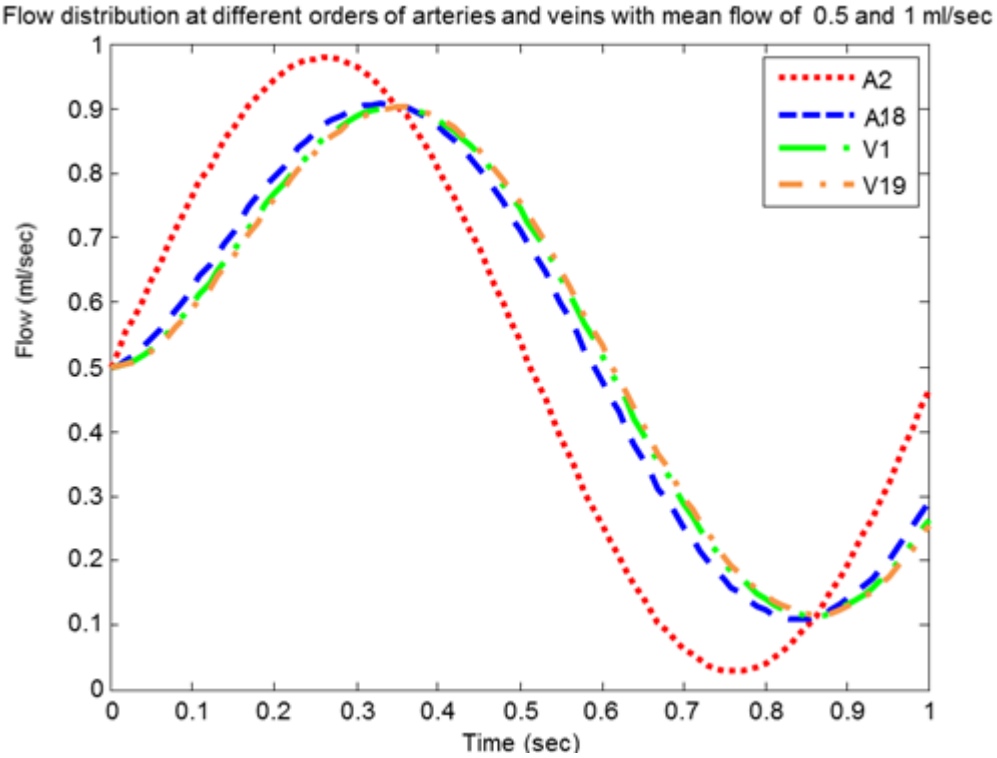
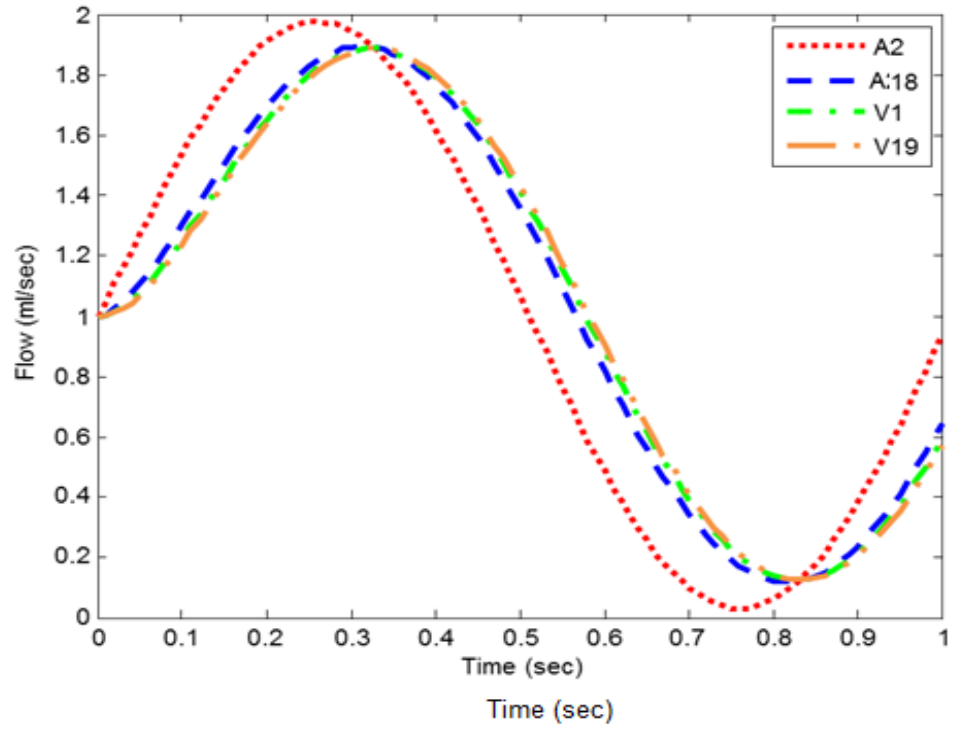


Figure 5.2: Model cardiac output with mean flow of 0.5 (top) and 1 (bottom) ml/sec.

Figure 5.3 shows model simulated flow at different orders of arteries and veins with mean flow of 0.5 (Top) or 1 (Bottom) ml/sec. In this plot A2 corresponds to arteries with diameter ranging from 230 to 177 microns and A18 corresponds to arteries with diameters 28 to 21 microns, V1 corresponds to veins with diameters 278 to 211 microns and V19 corresponds to veins with diameters 25 to 19 microns. For both flow cases, the calculated mean flow is constant across different orders, indicating that flow is conserved in the model.


Flow distribution at different orders of arteries and veins with mean flow of 0.5 and 1 ml/sec
A2
A18
V1
V19
Flow (ml/sec)
Time (sec)


A2
A18
V1
V19
Flow (ml/sec)
Time (sec)

Time (sec)

Figure 5.3: Calculated flow at different orders of arteries and veins with mean flow of 0.5 ml/sec (Top) and 1 ml/sec (Bottom). In the legend, A and V represent arteries and veins and the number represents the order.

5.1.3 Pressure vs. Flow as a function of Hematocrit

Changes in blood hematocrit (*Hct*) can have a substantial effect on pulmonary vascular resistance (Benis, Peslin, Mortara & Lockhart, 1967; Murray, Karp & Nadel, 1969). Thus, we used the model to calculate pressure - flow relationships over a range of hematocrit ratios ($Hct = 0 – 0.80$) using the parameters given in **Table 5.2**. In several studies, high blood hematocrit was reported in rats exposed to chronic hypoxia, and is now thought to be an important contributor to the observed increase in pulmonary arterial pressure (Barer, Bee, & Wach, 1983; Clarke, Baumgardt, & Molthen, 2010). **Figure 5.4** shows the calculated arterial - venous pressure drop at each flow for each value of *Hct*. As *Hct* increased, *Pa - Pv* increased with the greatest differences occurring at the highest *Hct* levels, this result is consistent with previously reported results (Barer, Bee, & Wach, 1983).

Input Parameters	Model Values
Mean Flow (ml/sec)	0 - 2.5
PA (cm-H_20)	0
Pv (cm-H_20)	3.5
Ppl (cm-H_20)	-3.5
Hct (ratio)	0 – 0.80

Table 5.2: Model parameter inputs to study the effect of blood hematocrit on calculated pressure- flow relationship.

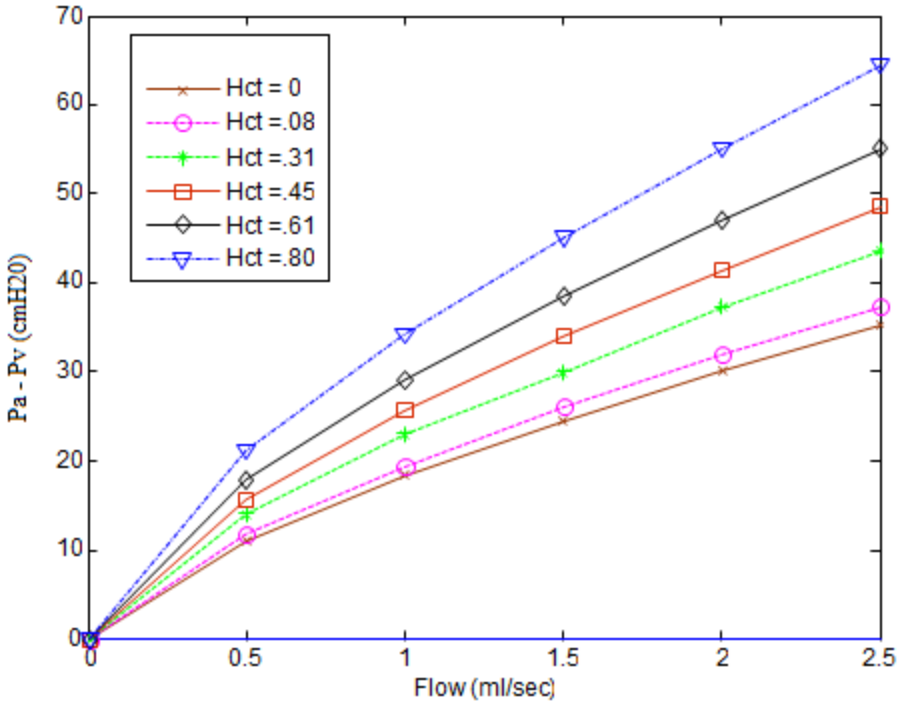


Figure 5.4: Calculated pressure - flow curves for different hematocrit (*Hct*) values.

5.1.4 Cumulative resistance vs. transpulmonary pressure

Previous studies have shown a U-shaped relationship between vascular resistance and transpulmonary pressure (P_{tp}) (Haworth S. T., 1996). A commonly accepted concept is that the extra-alveolar vessels distend and lengthen as P_{tp} increases, resulting in a decrease in vascular resistance. Concurrently, the distension of alveoli causes lengthening and narrowing of the alveolar vessels, increasing their resistance. The sum of the hemodynamic responses of these two vessel types, which are serially connected results in the characteristic U- shaped total vascular resistance curve. (Haworth S. T., 1996). Thus, we used the model to calculate cumulative resistance as a function of transpulmonary pressure. **Figure 5.5** shows the results obtained when venous pressure was varied from 0 to 14 cmH_20, using the model parameters of **Table 5.1**. We can see that as P_{tp} increases,

overall resistance (R) increases resulting in a U-shaped curve. This result is consistent with a compression of the capillary sheet (decrease in its diameter) causing increased resistance at higher P_{tp} values.

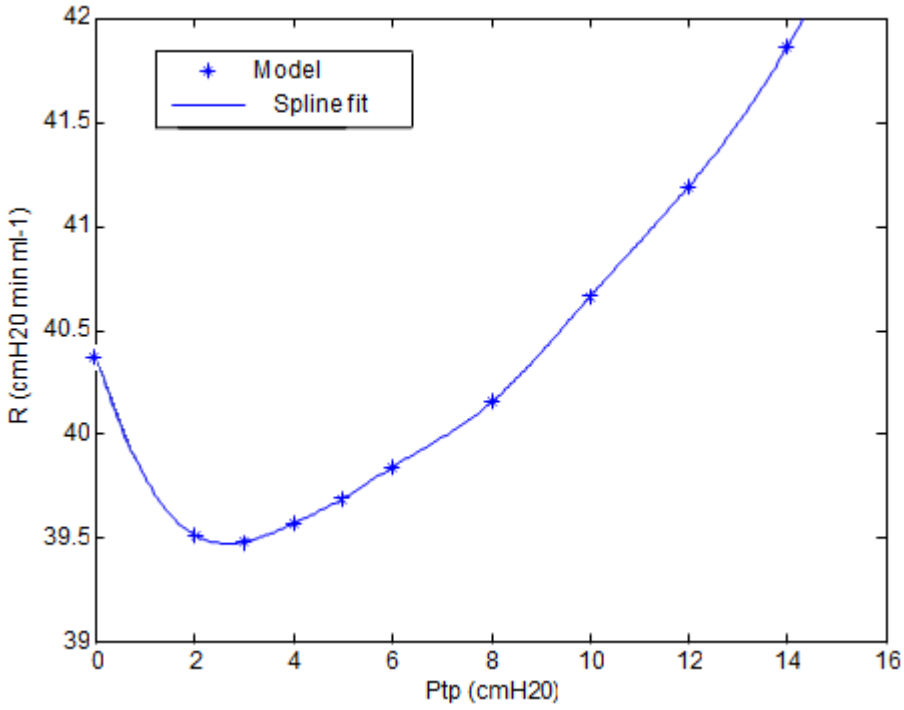


Figure 5.5: Vascular resistance versus transpulmonary (P_{tp}) pressure. Solid line is a spline fit to the model.

5.2 Model Validation

Prior to using the hemodynamic model to evaluate particular hypotheses, we validated the model by comparing the model output using the input values of **Table 5.1** with published experiment data.

Table 5.3 shows the resulting calculated pressures, volumes, resistances and compliances from the rat lung model. Here *Pa* and *Pc* represent mean pulmonary arterial

and capillary pressure; *Va, Vc, Vv* and *VL* represent the segmental arterial, capillary, venous and total volume; *Ca, Cc, Cv* and *Ra, Rc, Rv* represent the segmental arterial, capillary, venous compliances and resistances; and *CL* and *RL* are the total compliance and resistance of the lung.

Output parameters (Units)	Value
Pa (cm-H20)	16.60
Pc (cm-H20)	11.30
Va (ml)	0.272
Vc (ml)	0.551
Vv (ml)	0.302
VL (ml)	1.126
Ca (ml/cm-H20)	0.007
Cc (ml/cm-H20)	0.027
Cv (ml/cm-H20)	0.006
CL (ml/cm-H20)	0.041
Ra (cm-H20sec/ml)	8.710
Rc (cm-H20sec/ml)	7.688
Rv (cm-H20sec/ml)	10.07
RL (cm-H20sec/ml)	26.47

Table 5.3: Vascular pressures, segmental volumes, resistances and compliances calculated from the rat lung model.

The calculated model results are compared with experiment results from various pulmonary hemodynamic studies. Typical physiological pressure in the pulmonary artery of a rat is reported to be ~ 17.7 cm-H_20 (13 Torr) (Molthen, Wietholt, Haworth, & Dawson, 2004), whereas the model calculated *Pa* was relatively close 16.60 cm-H_20. The calculated model value of capillary pressure *Pc* = 12.28 cm-H_20 falls within the normal physiological range of 9 – 15 cm-H_20 (Presson, 1997).

We also compared calculated vascular volume in different regions of the vasculature to previously reported values. Total pulmonary vascular volume was prescribed to be 1.12 ml in our model given the input morphometric data describing vessel lengths, diameters and numbers and the capillary bed. This value is consistent with published values (Shifren, Durmowicz, Knusten, Hirano, & Mecham, 2007). Previous estimates of *capillary* blood volume in the rat are 0.66 ml (Crapo, Barry, Foscue, & Shelburne, 1980) and 0.48 ml (Weibel, 1970) measured using morphometric techniques. Molthen et al. reported estimates of capillary volume ranging from 44%-48% of total pulmonary vascular volume in small mammals and used the assumption that blood volume is split approximately in half between veins and arteries of the lung (Molthen, Wietholt, Haworth, & Dawson, 2004). When the model was run with the **Table 5.1** input values, the calculated values of arterial (0.27 ml), capillary (0.5 ml) and total volume (1.12 ml) fell within previously reported ranges from experimental data and modeling results.

Previous studies have shown that ~ 65% of the total pulmonary vascular compliance in rats was in vessels less than 40 microns, mostly in the capillaries (Presson, et al., 1998). The model calculated distribution of total vascular compliance in capillaries 67% which is consistent with reported results.

The segmental distribution of pulmonary resistance in rats was studied by arterial and venous occlusions (Alessandro, 2005). The reported *Ra*, *Rv*, and *Rc* represent about 36%, 22% and 42% of the total resistance respectively. Our model calculated distribution

of resistance of *Ra*, *Rv* and *Rc* is 33%, 29%, 38% respectively, which is consistent with the experiment results (Alessandro, 2005).

5.2.1 Distribution of vascular pressure, resistance, or compliance versus volume

Using the model results, the distributions of intravascular pressure, cumulative resistance (Rcum), cumulative compliance (Ccum) and cumulative inertance (Lcum) were calculated and graphed as a function of cumulative volume (Vcum) in **Figure 5.6** and **Figure 5.7**.

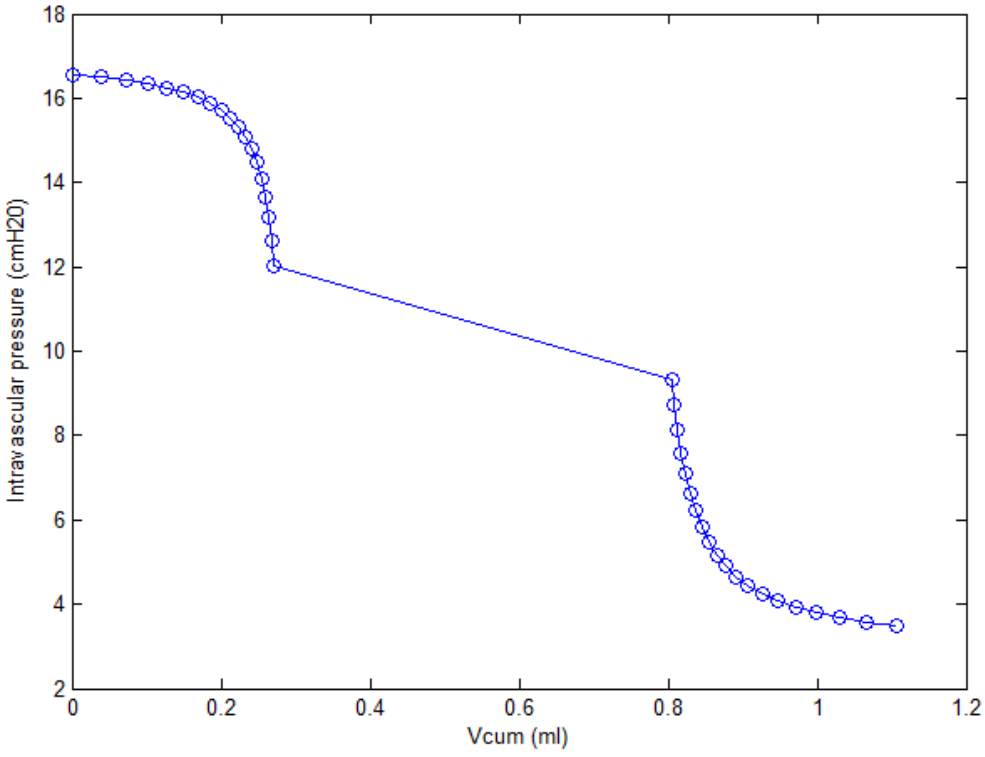


Figure 5.6: Intravascular pressure as a function of cumulative vascular volume (Vcum) from the rat lung model. On the Vcum axis, 0 is the inlet to the pulmonary artery, and a total volume of ~1.12 ml designates the exit from the pulmonary vein. Points indicate entrance to each vessel order so that the pressure drop within each order is the vertical

distance between successive points and the volume in each order is the horizontal distance between points. The long segment in the middle region represents the capillary sheet.

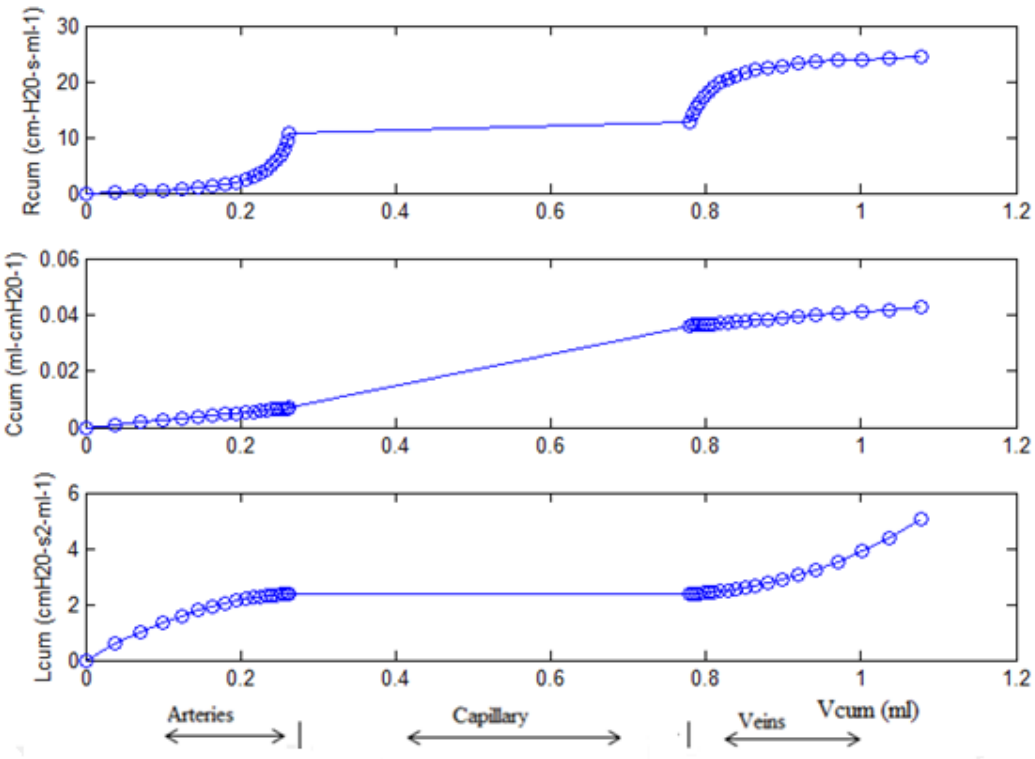


Figure 5.7: Cumulative vascular resistance (top), compliance (middle), and inertance (bottom) as a function of cumulative vascular volume. On the Vcum axis, 0 is the inlet to the pulmonary artery, and a total volume of ~1.12 ml designates the exit from the pulmonary vein. Points represent the inlet to each order of vessels.

Figure 5.6 shows intravascular pressure as a function of cumulative vascular volume (Vcum) in a rat lung model. On the Vcum scale, 0 is the inlet to the pulmonary artery, and a total volume of ~1.12 ml designates the exit from the pulmonary vein. The long segment in the middle region represents the capillary sheet. The concavity of the graph of the arterial and venous intravascular pressures shows that the larger arteries and veins contribute relatively little to the calculated total vascular resistance and are, therefore, responsible for only a small fraction of the model arterial-venous pressure

drop, which is consistent with previous data in dogs (Haworth, Linehan, Bronikowski, & Dawson, 1991). The distribution of the total vascular volume in arteries, capillaries and veins calculated from the model was 24.2%, 47.2%, 28.5%, which are in the normal range of experiment results (Molthen, Wietholt, Haworth, & Dawson, 2002).

The distributions of Rcum, Ccum and Lcum calculated from the model are graphed as a function of Vcum in **Figure 5.7**. The smaller arteries and veins impart a large fraction of the total resistance but contribute relatively little to the cumulative vascular volume (Haworth S. T., 1996). Calculated cumulative compliance is a nearly linear function of cumulative volume suggesting that, in comparison to the resistance distribution, the local vascular compliance per unit local vascular volume is relatively constant throughout the vascular bed. Thus there is a relative concentration of compliance in large arteries and veins (diameters > ~200 microns) and in the capillary sheet that is associated with the relatively large volume fractions in these portions of the model vascular bed (Haworth S. T., 1996). The calculated model inertance versus cumulative volume curve reveals that the larger diameter arteries and veins are the location of the most of the inertance as shown in the studies by Haworth et.al (Haworth, Linehan, Bronikowski, & Dawson, 1991).

Dynamic Simulation Results

For dynamic simulations, the sinusoidal cardiac output was set at a heart rate of 7 beats/sec and a systolic/diastolic ratio of 0.9, with a breathing rate of 1.2 breaths/sec, all typical for a rat. The mean values of pulmonary arterial pressure and capillary pressure are those of **Table 5.3**. The parameters values used to run the dynamic simulation are

given in **Table 5.1**. The minimum and maximum transpulmonary pressure used to run this simulation was 3.5 and 4 cmH_20. First the dynamic model was run over a 5 second interval with a time step of 0.01 seconds with the constant *RLC* option The quasi- steady state is determined in the model, by checking if the mean pressures and flows remain constant over a period of 5 seconds, and in this case reached after ~ 2 seconds. The calculated dynamic results for pressures and flows are shown in **Figure 5.8** and **Figure 5.9**.

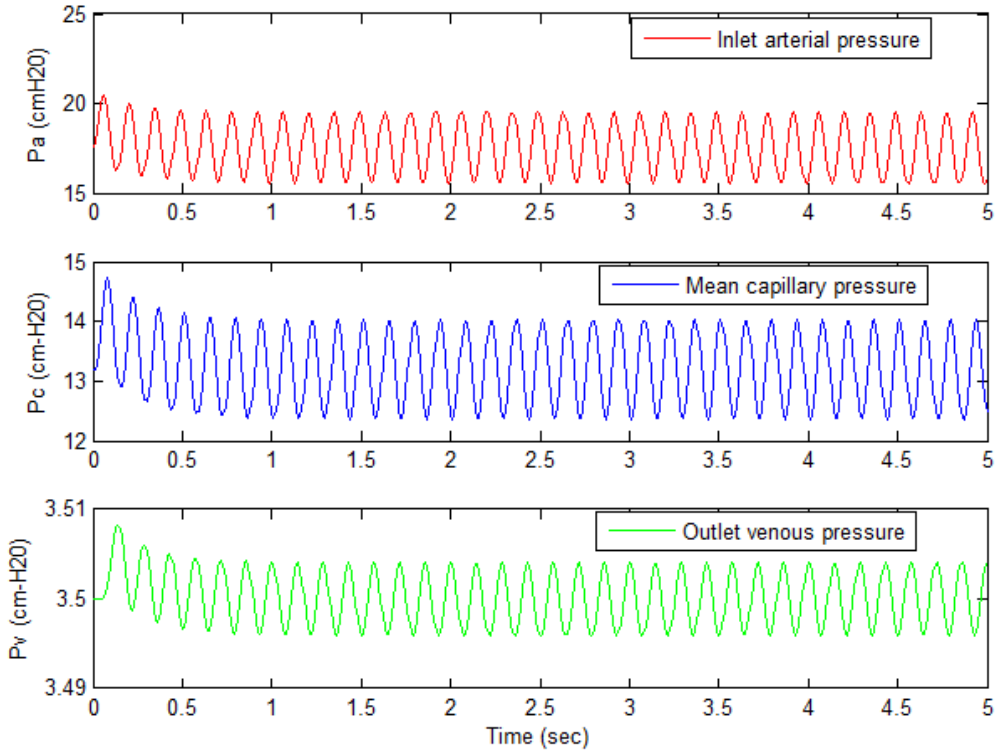


Figure 5.8: Constant *RLC* simulation: Calculated pressure at the inlet to the pulmonary arterial tree (red), the mean capillary pressure (blue), and the outlet venous pressure (green) over 5 seconds.

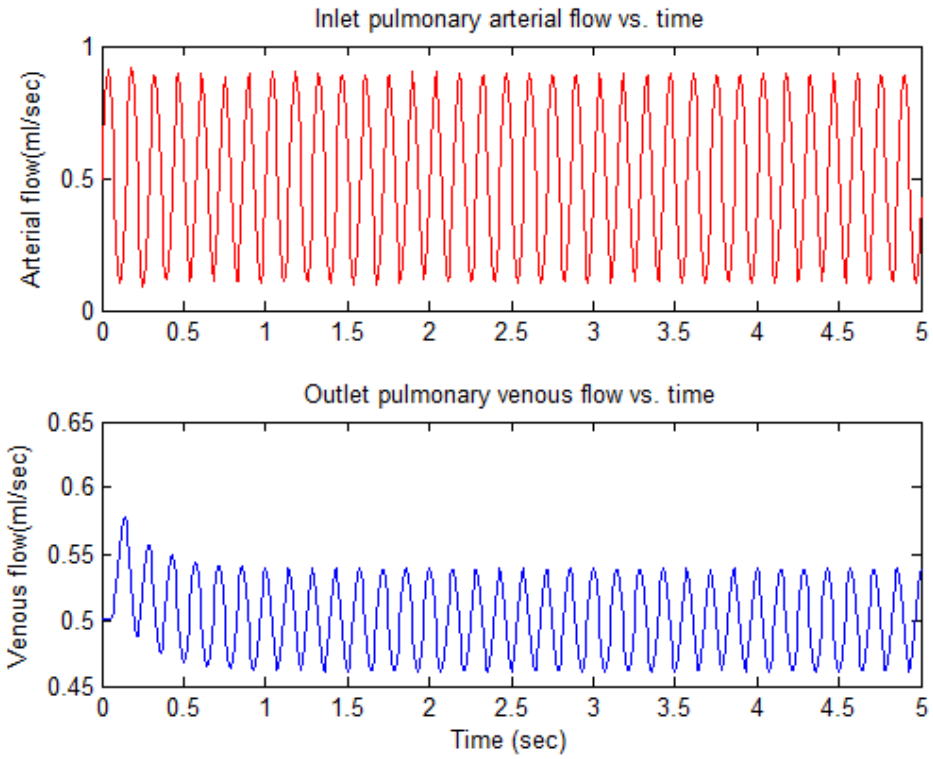


Figure 5.9: Constant *RLC*: Calculated flow at the inlet to the pulmonary arterial tree (red) and venous outlet (blue) over 5 seconds.

The simulation was repeated for a 1 minute interval using the variable *RLC* option with a time step of 0.01 seconds. In the variable *RLC*, the values of *R*, *L* and *C* are updated at every time step and the model results reached a quasi-steady state after ~30 seconds. The calculated dynamic pressures and flows for this case are shown in **Figure 5.10** and **Figure 5.11**.

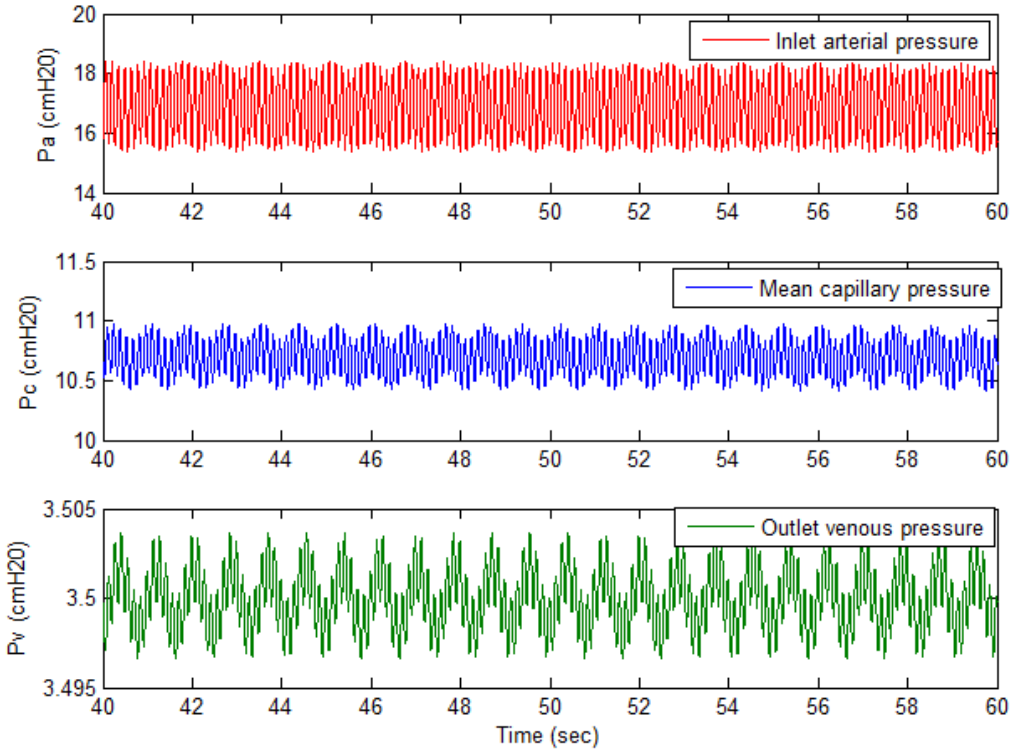


Figure 5.10: Variable *RLC*: Calculated pressure at the inlet to the pulmonary arterial tree (red), the mean capillary pressure (blue), and the outlet venous pressure (green) over the 40 to 60 seconds.

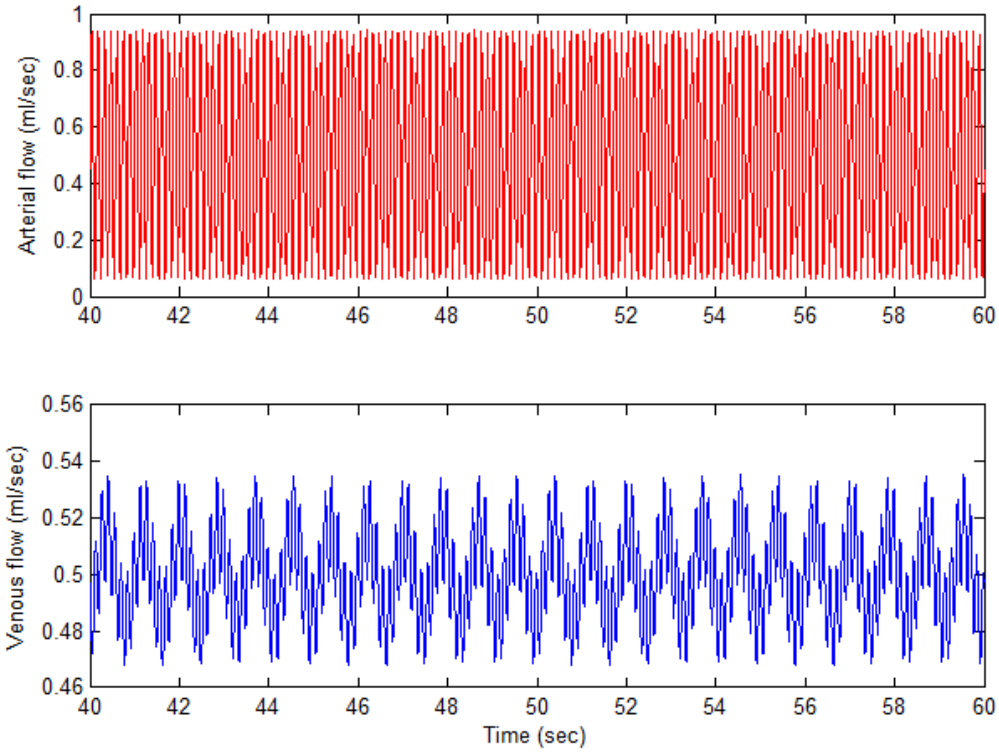


Figure 5.11: Variable *RLC*: Calculated flow at the inlet to the pulmonary arterial tree (red) and venous outlet (blue) over 40-60 seconds.

5.2.2 Pressure versus Flow at different airway pressures

Experiment 1

We also validated the model by comparing the calculated model output results with pilot experimental arterial pressure versus flow data obtained in our laboratory by Dr. Robert Molthen using the isolated perfused lung preparation described below. All procedures were approved by the Institutional Animal Care and Use Committees of the Zablocki VA Medical Center.

An adult male Sprague-Dawley rat was anesthetized (sodium pentobarbital, 40 mg/kg ip) and the lungs removed. The trachea was cannulated and connected to a

ventilator; the pulmonary artery was cannulated and connected to a perfusion pump in a recirculating system. The cannulated isolated lung was then suspended from the perfusion apparatus for subsequent manipulations and measurements. The lung was ventilated with a 15% O_2, 6 % CO_2 in N_2 gas mixture, 3 mmHg end expiratory pressure and 8 mmHg end inspiratory pressure. The lung was perfused with a physiological salt solution (perfusate) containing 5% bovine serum albumin. The vasodilator papaverine hydrochloride (0.6 mg/ml) was added to the reservoir of perfusate and circulated for approximately two minutes (Molthen, Karau, & Dawson, 2004). With the tracheal pressure set at 3, 6, 10, or 12 mmHg, hemodynamic perfusion studies were performed at flow rates of 0, 5, 10, 20, 30 and 40 ml/min. The experiment set up is shown in **Figure 5.12**. The values of the key experimental parameters are shown in **Table 5.4**.

Figure 5.12: Experiment set up for the isolated lung perfusion.

Parameters (Units)	Experiment Value
Mean flow (ml/min)	0 – 40
Pv (mmHg)	0
Ppl (mmHg)	0
PA (mmHg)	3-12
Hct (ratio)	0

Table 5.4: Experiment and model input parameters and values for the pressure-flow data (Molthen.et.al 2004).

Model Simulation Methods

The experiment parameters given in **Table 5.4** were used as input to the model and the resulting simulation output is shown in **Figure 5.13** for airway pressures ranging from 3 to 12 mmHg. In this case, it is clear that the model results are not in agreement with the experimental data. In the model simulation, *Pv* was set to 0, resulting in zone 2 conditions (*PA* > *Pv*). Thus we made the following adjustments to the model parameters.

i) *Pv* was set to 1.4 mmHg instead of 0. This is the critical closing pressure reported by Molthen et al. (Molthen, Haworth, Gordon, Krenz, & Clough, 2005) for a control rat and corresponds to the pressure at which the pulmonary vein completely collapses.

ii) The model airway pressure was set to half of the experiment value in order to compensate for the hydrostatic pressure gradient within the lung due to gravity, which is not accounted for in the model. Because of this gradient, the vascular pressure varies between the top and bottom of the upright lung, and the bottom of the upright lung receives proportionately more flow that the top of the lung.

The model simulation results under the above assumptions are shown in **Figure 5.14**. The model calculated pulmonary arterial pressure for each flow was compared to the control rat experiment data. The coefficient of variation (*cv*) between the experimental and model values of *Pa* was calculated as

$$cv = \frac{\sigma}{\overline{X}}.$$

where $\overline{X}$ is the mean difference between the experimental and model values and σ is the standard deviation about that mean.

The *cv values* between the model simulations with half the airway pressure and the experimental pressure-flow data at each of the airway pressures are given in **Table 5.5**. The *cv* was calculated excluding the data at flow at 0 ml/min.

Experiment PA (EPA) (mmHg)	Model PA = Half of EPA (mmHg)	*cv*
3	1.5	0.60
6	3	0.24
10	5	0.60
12	6	0.55

Table 5.5: Coefficient of variation calculated for *Experiment 1* for different airway pressures.

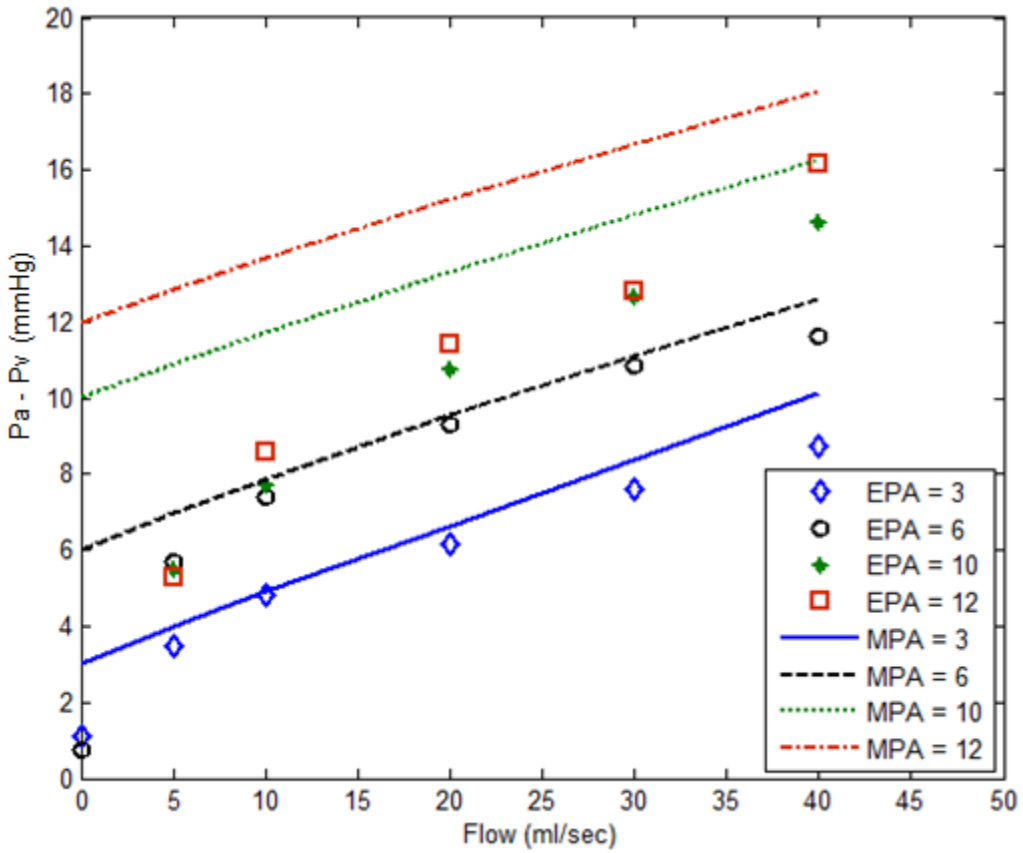


Figure 5.13: Difference between pulmonary arterial and venous pressure versus flow at different airway pressures (*PA*) with *Pv* = 0 mmHg, *Ppl* = 0 mmHg and *Hct* = 0. Symbols (E) represent experimental data while solid lines (M) are model results.

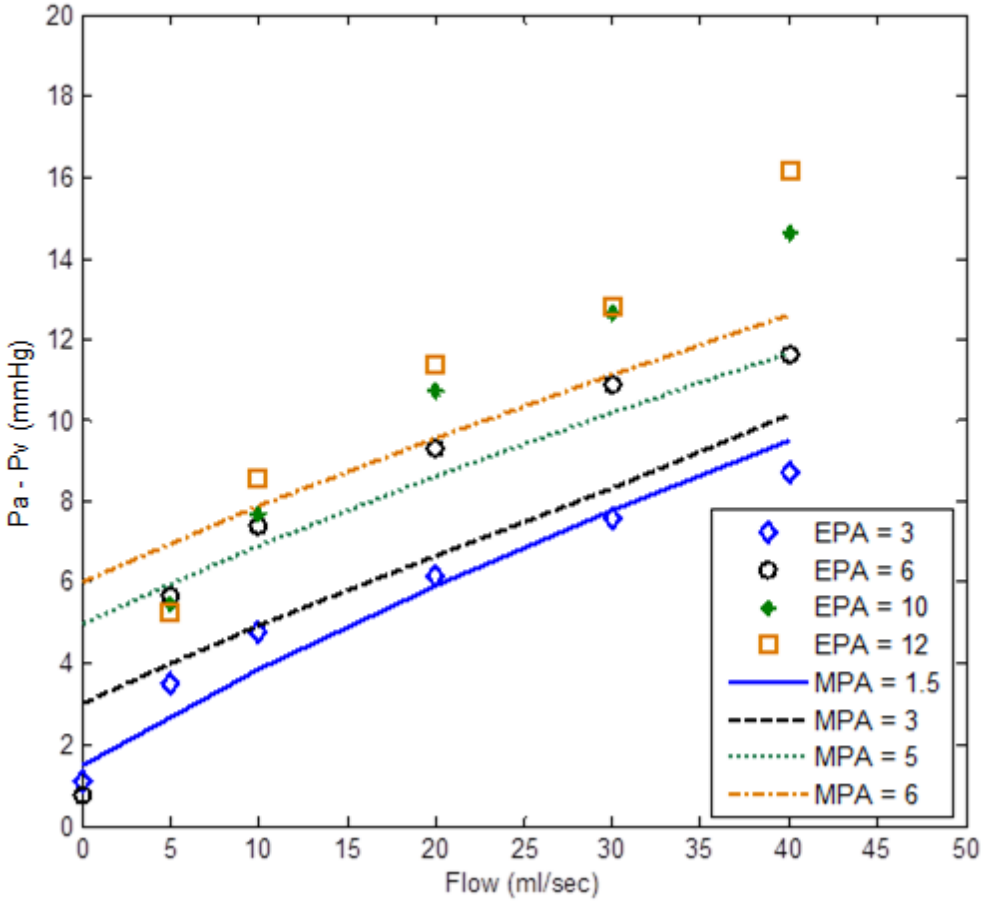


Figure 5.14 : Difference between pulmonary arterial and venous pressure versus flow at different airway pressures (*PA*) with *Pv* = 1.4 mmHg, *Ppl* = 0 mmHg and *Hct* = 0. Symbols (E) represent experimental data while solid lines (M) are model results. Here, *PA* is exactly half the experiment values.

Experiment 2

To further investigate the reason behind the difference between experimental data and model results, the same experimental protocol described above (Experiment 1) was repeated by Drs. Haworth and Audi using airway pressures ranging from ~1 to 6 mmHg. **Figure 5.16** shows experimental data and model simulations with *Pv* set at 1.4 mmHg (critical closing pressure) and the airway pressure set at half the experimental airway

pressure. The resulting model simulations captured the trend of the data quite well, as shown in **Figure 5.15**.

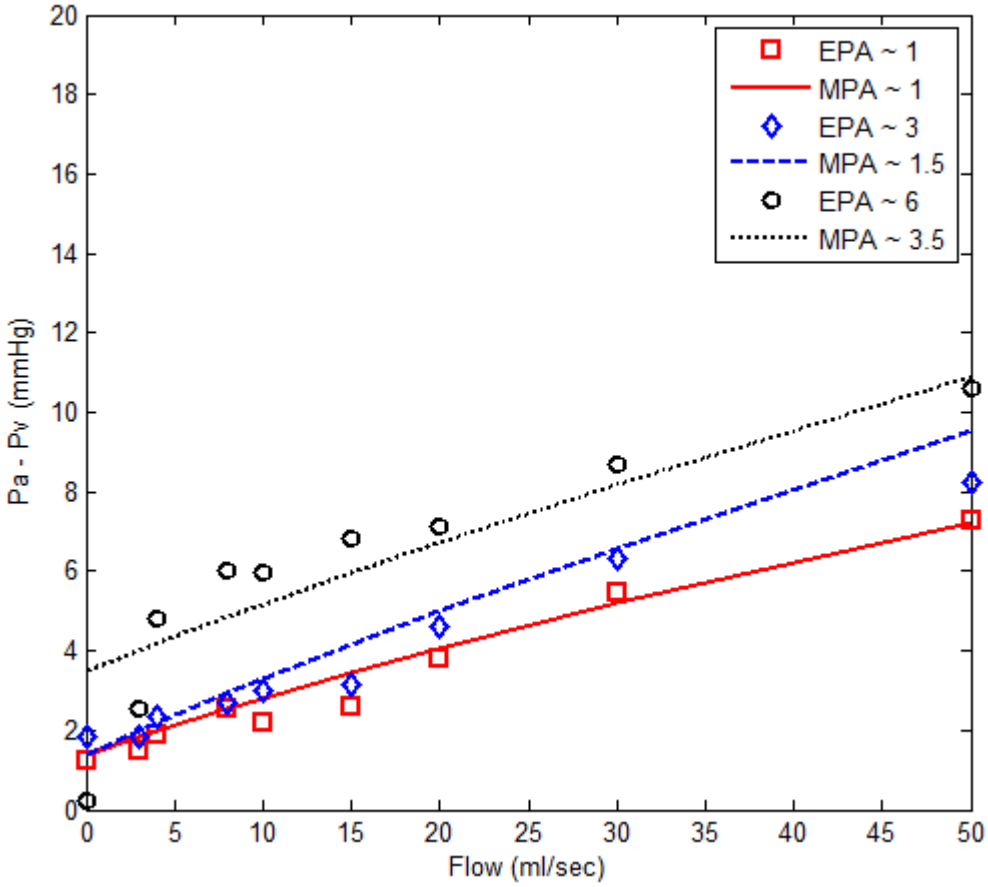


Figure 5.15: Difference between pulmonary arterial and venous pressure versus flow with different airway pressures with *Pv* =1.4 mmHg, *Ppl* = 0 mmHg, *Hct* = 0 and *PA* ~ half of the experiment values. E represents the preliminary experiment data and M represents the model results.

When *PA* ~ 1 mmHg, and *Pv* is set to be 1.4 mmHg in the model simulation, the lung is in zone 3 (**Figure 3.7**), i.e. *Pa* > *Pv* > *PA*. Under these conditions, the model appears to capture the trend of the data. At airways pressures > 1mmHg, the model appears to capture the trend of the data for non-zero flows. Note: for airway pressures of 3 mmHg and 6 mmHg, regions of the upright lung (top) are presumably in zone 2

(**Figure 3.7**) since, *Pa* > *PA* > *Pv.* In zone 2, the pressure gradient determining blood flow is the arterial- alveolar pressure difference. In this model, the finite thickness functional model is used to represent the zone 2 conditions, which was developed by Fry (Fry, Thomas, & Greenfeild, 1980) and later extended by Dawson et al. (Dawson, Rickaby, & Linehan, 1986). This model appears insufficient to adequately predict arterial pressure under these conditions, since it does not account for the hydrostatic pressure gradient within the upright lung.

6. MODEL INVESTIGATION

We used the rat lung model to investigate previously proposed factors responsible for the increase in pulmonary arterial pressure (*Pa*) observed in rats exposed chronically to hypoxia. Three factors postulated to be responsible increased *Pa* are

i) Decrease in arterial and venous distensibility (Molthen, Karau, & Dawson, 2004; Molthen, Gordon, Krenz, & Clough, 2007).
ii) Decrease in capillary sheet surface area (Molthen, Heinrich, Haworth, Krenz, & Gordon, 2004).
iii) Rarefaction of arteries (Rabinovitch, Gamble, Nadas, Miettinen, & Reid, 1979; Zhao, 2011; Hislop & Reid, 1976).

As described below, the effect of decreasing arterial and venous distensibility and the capillary sheet surface area on *Pa* were evaluated by changing the corresponding parameter values in the rat model to those measured in actual rat experiments. The effect of arterial rarefaction on *Pa* was evaluated by pruning small arteries from the model arterial tree. The model simulation results were used to evaluate whether one of these factors or a combination of them is responsible for the increase in *Pa*, by comparing the results with the experiment data.

6.1 Experimental Methods

The data and experimental methods described in this thesis are the work of Dr. Robert Molthen and have been previously published (Molthen, Karau, & Dawson, 2004).

We present an overview in order that the model simulations can be more readily appreciated relative to the previously published experimental data. All procedures were approved by the Institutional Animal Care and Use Committee of Zablocki VA Medical Center, Milwaukee, WI.

Animal exposures

One group of Sprague-Dawley rats (n = 9) 55-65 days old (250-350 gm.) was exposed to hypoxia using an environmental chamber comprised of an airtight enclosure large enough to house a standard rat cage (Molthen, Karau, & Dawson, 2004) and ports for inflow and exhaust gases. To produce normobaric hypoxia, a mixture of room air and nitrogen was pumped through the chamber in order to maintain an oxygen concentration of 10%. Total flow delivery to the chamber was 3-4 liters/minute. The second group (n = 9) was housed under similar, but normoxic (room air) conditions (Molthen, Karau, & Dawson, 2004). After 21 days, each rat was anesthetized (sodium pentobarbital, 40 mg/kg ip), and the lungs removed and prepared as described above in Section 5.2.2.

Hemodynamic perfusion studies

Pressure – flow data was acquired at flow rates of 5, 10, 20, 30 and 40 ml/min with a tracheal pressure of 6 mmHg.

Lung micro-CT Imaging

The perfusate in the arterial tree was then replaced by filling it with perfluorooctyl bromide (perflubron, an intravascular x-ray contrast agent) at a tracheal pressure of 6 mmHg. At these pressures, perflubron tends not pass through the capillaries, therefore only the arteries fill with contrast medium. The lungs were then rotated in the x-ray beam

for CT imaging. This CT scanning was repeated at intravascular pressures of 5.4, 12, 21 and 30 mmHg using a height adjustable reservoir. The reconstructed CT images were used to obtain measures of vessel diameters at each intravascular pressure (Molthen, Karau, & Dawson, 2004). Then vessel distensibility was calculated by measuring the change in segment diameters along the arterial trunk with respect to changes in intravascular pressure.

6.2 Experimental Results

Here we summarize the key results reported by Molthen et al. (Molthen, Karau, & Dawson, 2004). The hematocrit measured in the hypoxic group was significantly higher than controls, (69 compared with 41 for normoxic rats). The combination of higher pulmonary vascular resistance and increased viscosity led to an increased workload for the right heart, reflected by a significant amount of right ventricular hypertrophy, as measured by the weight ratio of the dissected right ventricle to the remaining left ventricle plus septum, 0.572± 0.043 (SE, n = 9) in the hypoxic group compared with 0.267 ± 0.008 (n = 9) in the control group. By normalizing the right and left heart components to body weight, the authors concluded that there was no general cardiomegaly, because chronic hypoxia caused only the right ventricle to be larger. There was also a significant increase in dry lung weight in the hypoxia-exposed rats compared with the control group, 0.322 ± 0.010 (SE, n = 9)g vs. 0.267 ± 0.018 (n = 9) g, both before and after normalizing for body weight.

Figure 6.1 shows pulmonary arterial minus venous pressure as a function of flow obtained from the isolated lung experiments previously reported (Molthen, Karau, &

Dawson, 2004). The lungs of the hypoxia-exposed rats exhibited increased perfusion pressure compared with normoxic controls. From these data, they estimated the values of arterial distensibility (α_A) for lungs of hypoxic and normoxic rats to be 0.015 ± 0.003 (SE) (n = 8) and 0.028 ± 0.001 (n = 9) $mmHg^{-1}$, respectively, a significance difference ($p < 0.001$).

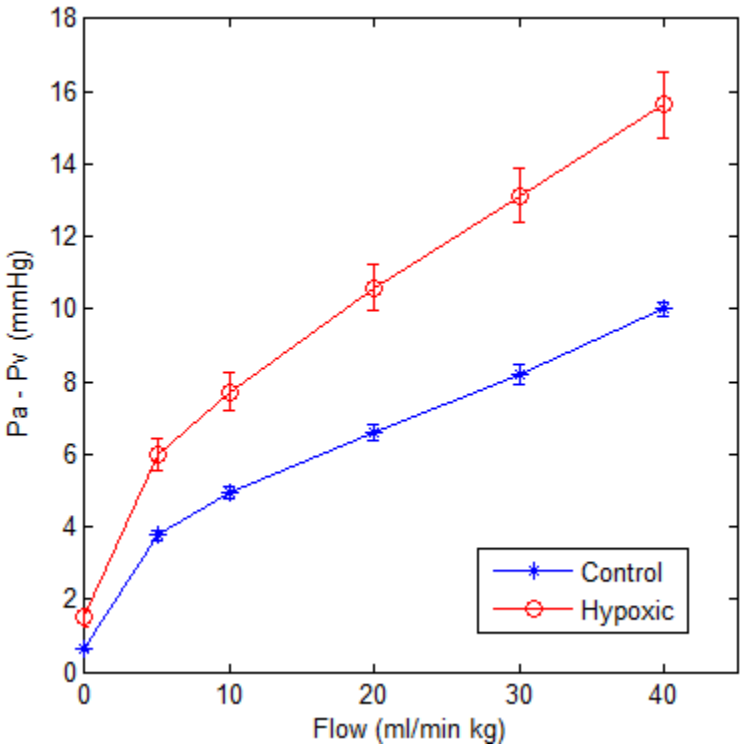


Figure 6.1: Difference between pulmonary arterial and venous pressure vs. flow in isolated lungs from normoxic control (n = 9) and hypoxia-exposed (n = 6) rats (mean ± SE). Flow is normalized to body weight.

6.3 Model Simulation Methods

We proceeded to use the rat model to simulate the same experimental conditions and then compared the pressure versus flow data from the experiment and the model simulation results. **Table 6.1** shows the values of the steady state inputs used in the

experiment and model simulation for a control rat. Again, we compensated for the hydrostatic pressure gradient within the upright lung by setting the airway pressure to 3 mmHg (instead of 6 mmHg). The venous pressure was set to 1.4 mmHg (the critical closing pressure), as explained in Section 5.2.2.

Steady state parameters	Experiment	Simulation
Flow (ml/min)	0 - 40	0 – 40
Venous pressure (mmHg)	0	1.4
Pleural pressure (mmHg)	0	0
Airway pressure (mmHg)	6	3
Hematocrit	0	0

Table 6.1: Steady state parameter values used in the experiment and the model simulation.

Initially, the most general functional models (i.e. those with the greater number of free parameters) were selected in order to provide as much flexibility as possible to the simulated pressure-flow curves. These models are specified in **Table 6.2**.

Functional Models	Option
Arterial and venous vessel distension	Nonlinear (Linehan, F.deMora, Bronikowski, & Dawson, 1988)
Capillary sheet distension	Nonlinear (Glazier, Hughes, Maloney, & West, 1969)
Perivascular pressure	Haworth and Smith/Mitzner (Smith & Haworth, 1998)
Lung air volume	Deflation (Haworth S. T., 1996)
Arterial and venous length vs. volume	Smith/Mitzner (Smith & Mitzner, 1980)
Capillary sheet length and width vs. volume	Smith/Mitzner (Smith & Mitzner, 1980)
Capillary post diameter vs. volume	Constant volume (Haworth S. T., 1996)
Capillary Zone 2 behavior	Finite minimum thickness model (Fry, Thomas, & Greenfeild, 1980)
Apparent viscosity	Kianihudetz (Kiani & Hudetz, 1991)

Table 6.2: Functional model options used in the model simulation.

With these settings, the model was run with flow rates varying from 0 – 40 ml/min as in the experiment. The model calculated pulmonary arterial pressure for each flow was compared to the control rat experiment data. The coefficient of variation (*cv*) between the experimental and model values of *Pa* was calculated.

In order to determine whether a simpler reduced model form with fewer parameters might be suitable, the simulations were rerun by varying each of the functional model selections independently. However, in each case the resulting *cv* was higher with the reduced model leading us to settle on the functional models given in **Table 6.2**.

6.4 Model Simulation Results

We proceeded to compare the model simulation results with the measured experiment results obtained under control and hypoxic conditions by systematically varying the arterial and venous distensibility values, reducing the capillary sheet area and reducing the number of arterial vessels and a combination of all these factors.

6.4.1 Control conditions

Figure 6.2 shows the best fit of the model with the experiment data which gives the least *cv*. The steady state parameters and functional model options for the control condition is shown in **Table 6.1** and **Table 6.2**. Note that here we neglected flow at 0 ml/min in when calculating the *cv*. The resulting coefficient of variation between the experimental and model *Pa* results was 0.25.

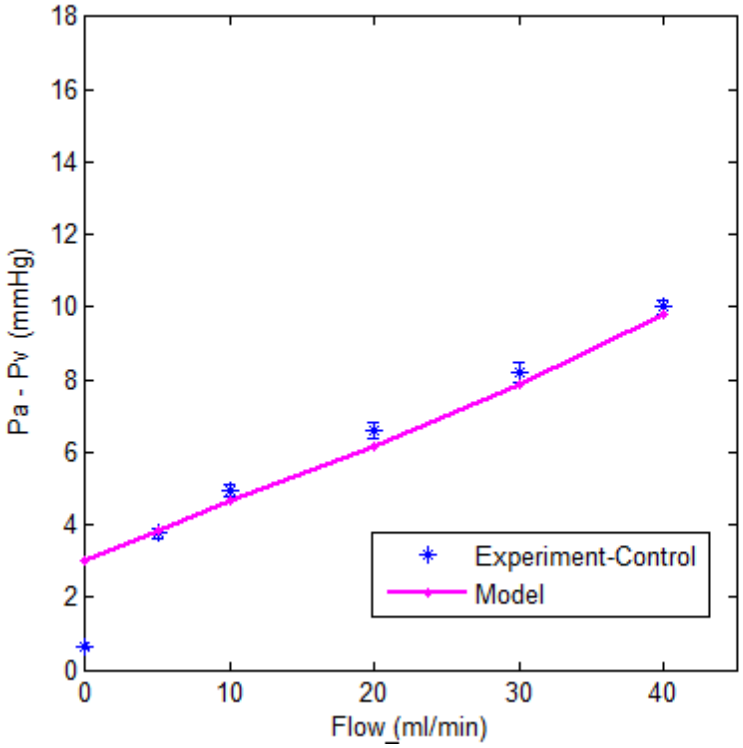


Figure 6.2: Difference between pulmonary arterial and venous pressure vs. flow for control rats. Symbols represent experimental data whereas solid line represents the optimal model result.

6.4.2 Effect of changing vessel distensibility

Stiffening of blood vessels is proposed to be one of the factors responsible for the observed increase in the pulmonary arterial pressure in chronic hypoxia (Zhao, 2011; Molthen, Karau, & Dawson, 2004). Thus, the effect of changing vessel distensibility on pulmonary arterial pressure was studied by first changing the arterial distensibility, then the venous distensibility and then both arterial and venous distensibility in the model. **Table 6.3** shows the reported experiment values for arterial (Molthen, Karau, & Dawson, 2004) and venous distensibility (Molthen, Gordon, Krenz, & Clough, 2007).

Distensibility (%/mmHg)	Control	Hypoxia
Arteries	2.4	1.3
Veins	2.0	0.2

Table 6.3: Control and hypoxic vessel distensibility model values for arteries and veins obtained from the experiment.

a. Effect of changing arterial distensibility (α_A)

Arterial distensibility (α_A) values was varied from 1 to 10 %/mmHg and its effect on pressure was simulated as shown in **Figure 6.3**. In these simulations, the venous distensibility value was set equal to the control value as shown in **Table 6.3**.

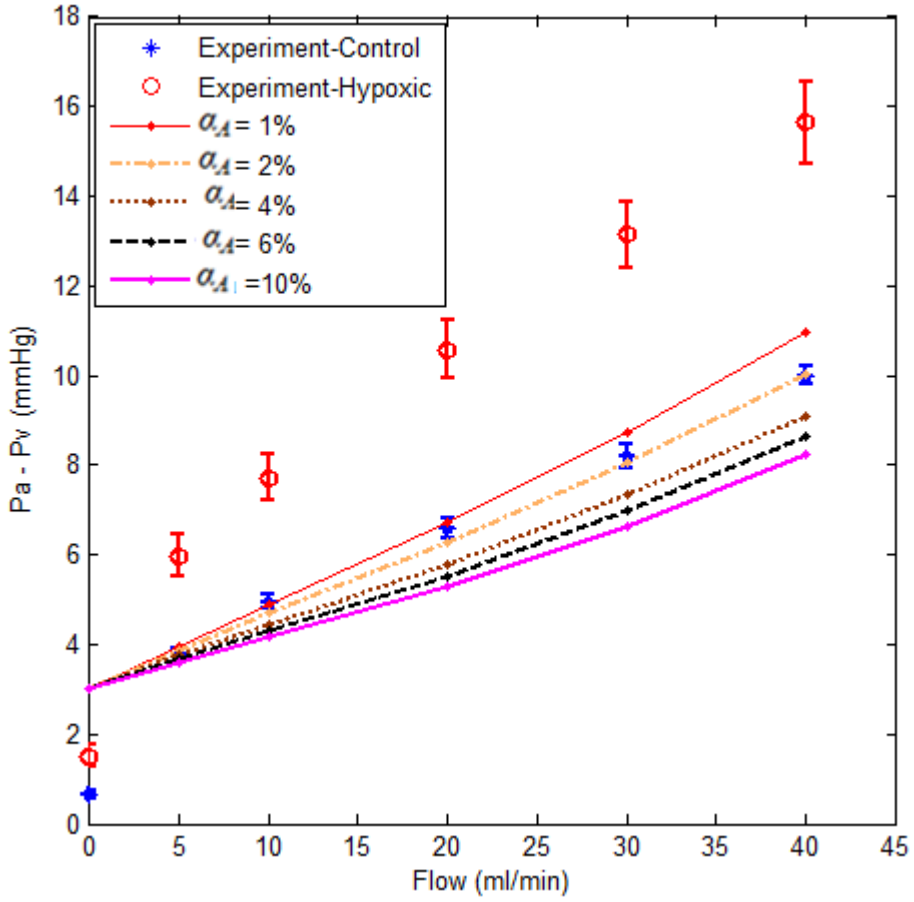


Figure 6.3: Difference between pulmonary arterial and venous pressure versus flow from model simulations using a range of values of arterial distensibility, α_A = 1 to 10 %/mmHg**.** Control and hypoxic experimental data are indicated by symbols.

b. Effect of changing venous distensibility (α_V)

Similarly, venous distensibility (α_V) was varied from 0.1 to 10 %/mmHg and its effect on the pressure versus flow curves was studied. **Figure 6.4** shows the resulting control pressure-flow graphs. In these simulations, the arterial distensibility value was set equal to the control value as shown in **Table 6.3**. Note that there was only a modest change in the pressure-flow curves when changing α_V values compared to changing α_A values over this range.

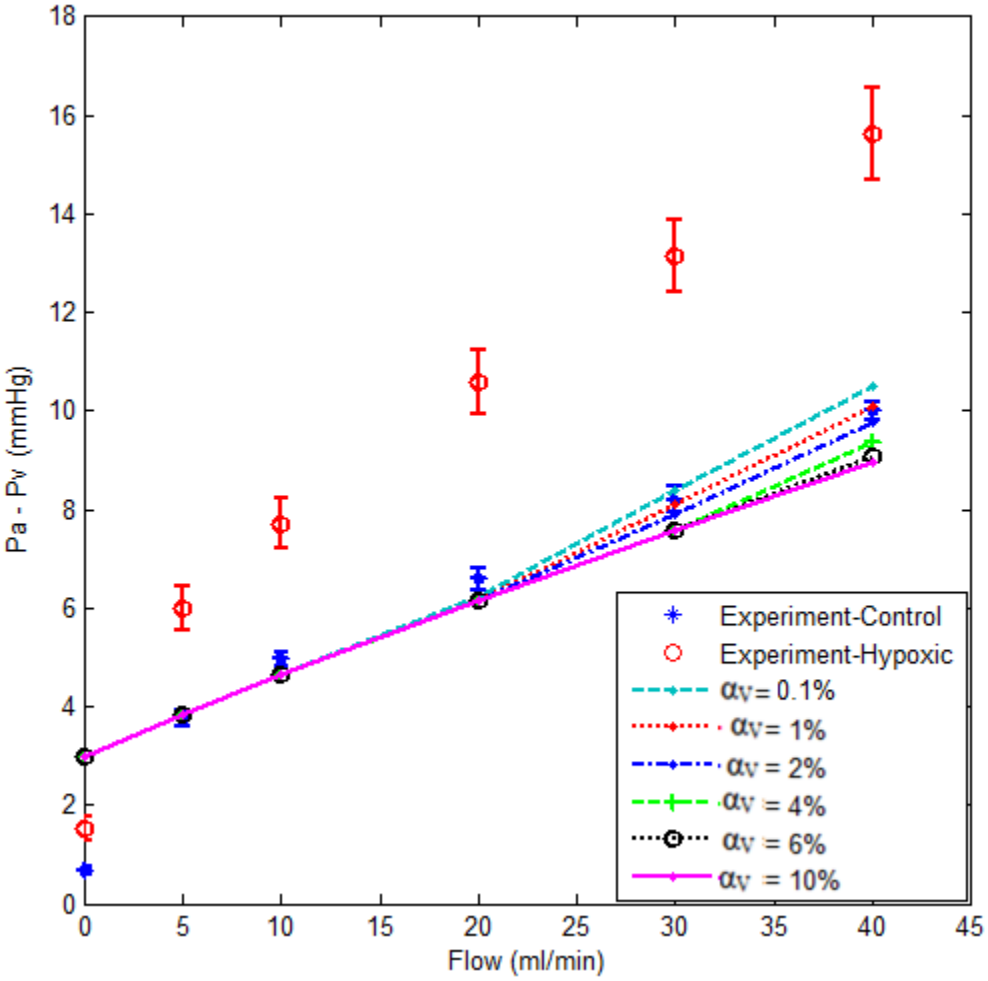


Figure 6.4: Model simulation result of difference between pulmonary arterial and venous pressure versus flow showing the effect of changing the venous distensibility ranging from 0.1 to 10%/mmHg.

c. Effect of changing arterial and venous distensibility

Finally, the arterial and venous distensibility values were changed to the reported hypoxic rat experiment values shown in **Table 6.3** and the resulting model pressure versus flow curve is shown in **Figure 6.5**.

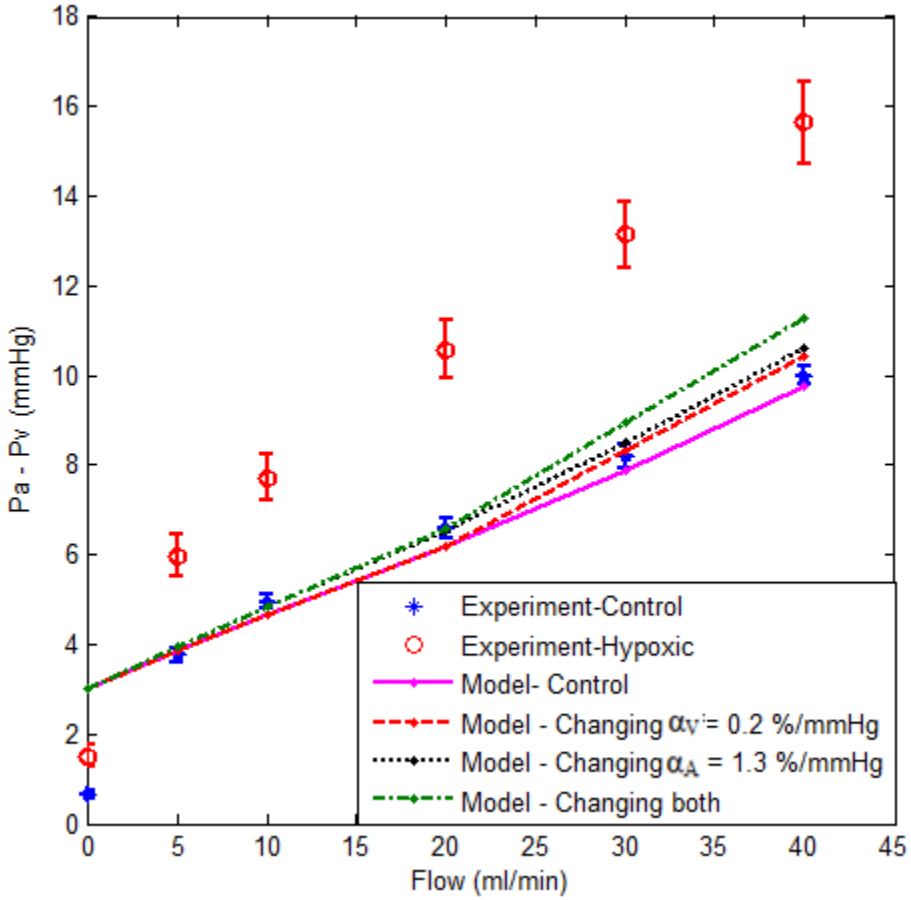


Figure 6.5: Model simulation result of difference between pulmonary arterial and venous pressure versus flow using showing the effect of changing the arterial distensibility, venous distensibility and both to the values in Table 6.3.

6.4.3 Effect of changing capillary sheet area

A reduction in capillary sheet area by 15% was also reported by Molthen et al. in rats exposed to chronic hypoxia (Molthen, Heinrich, Haworth, Krenz, & Gordon, 2004). We used the model to evaluate the impact of this reduction on pulmonary arterial pressure by reducing the control rat capillary sheet area (*CSA*) of 0.125 cm^2 by 15%, 30% and 45%. All other steady state parameters, functional model options and distensibility parameters were held the same as those of the control rat simulation (**Table 6.1**, **Table**

6.2 and **Table 6.3**). **Figure 6.6** reveals that there was only a modest increase in pressure when *CSA* was reduced by the reported 15%, under these conditions.

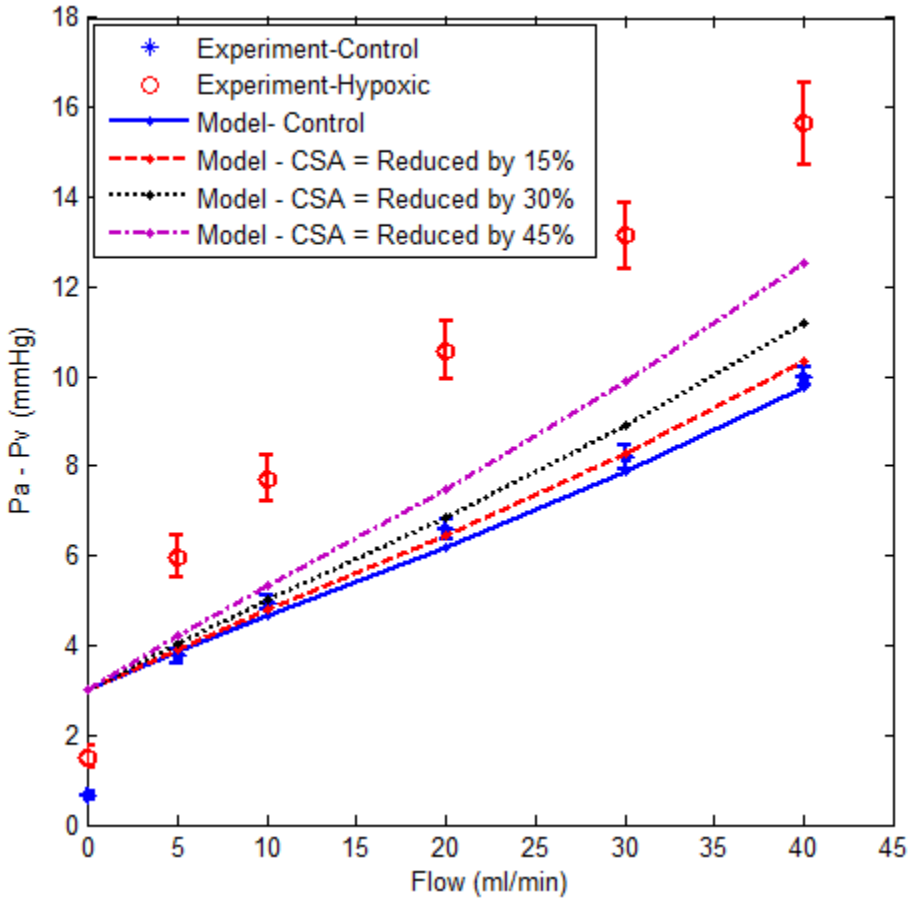


Figure 6.6: Model simulation result of the effect of reducing capillary sheet area (*CSA* = 0.123 cm^2) by 15, 30 and 45 %.

6.4.4 Effect of arterial rarefaction

Hislop and Reid (Hislop & Reid, 1976) reported that in hypoxic rats, the microscopic counts of small arteries showed that vessels smaller than 200 μm external diameter were gradually "lost". They reported a loss of ~ 38% of arterial vessels less than 200 μm in rats exposed to hypoxia for 21 days. Thus, we simulated the effect of arterial

rarefaction on pulmonary arterial pressure using the model by eliminating a specified percentage (15%, 30%, 38%, and 60%) of arteries with diameters less than 200 microns. **Figure 6.7** shows the resulting change in the pressure-flow curves compared to the experimental control and hypoxia rat data and shows that arterial rarefaction appears to have the greatest impact on pressure, relative to the other effects simulated thus far.

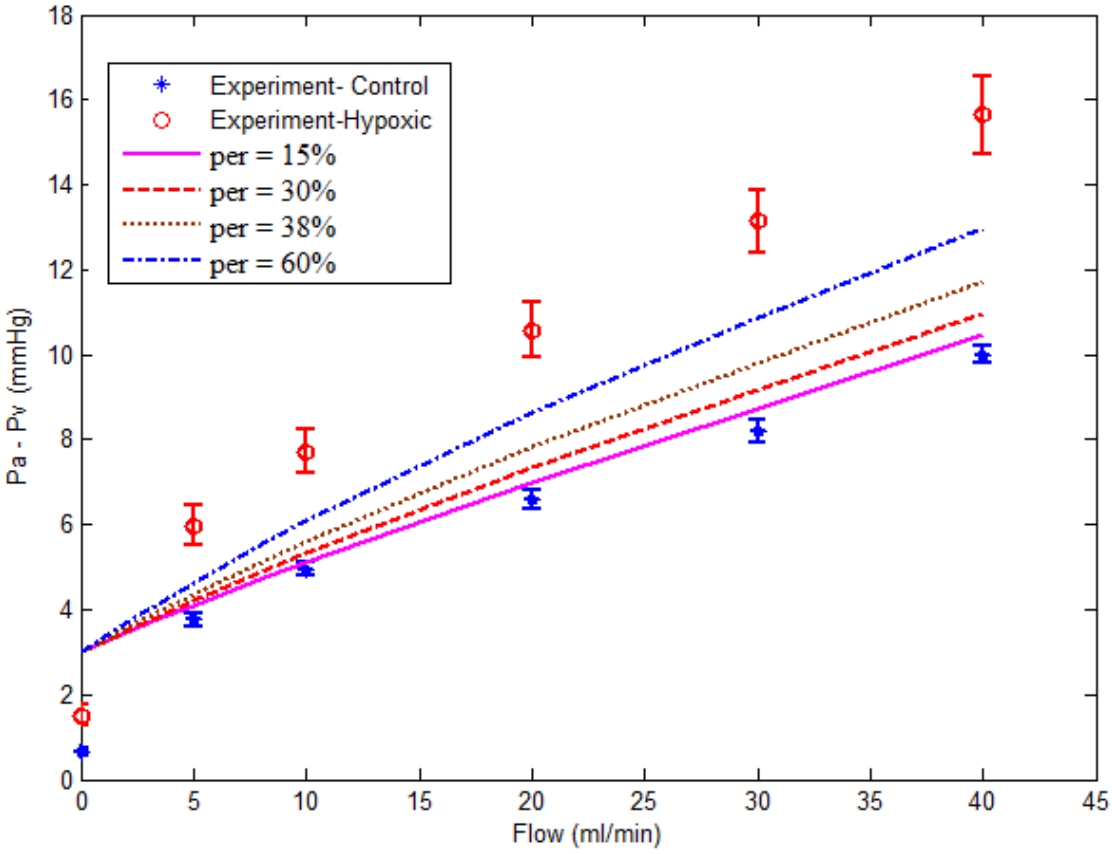


Figure 6.7: Model simulation result of difference between pulmonary arterial and venous pressure versus flow showing the effect of removing 15% - 60% of arteries smaller than 200 micron diameter.

The effect of eliminating a percentage (15%, 30%, 38%, and 60%) of arteries with diameters greater than 200 microns on the pressure is shown in **Figure 6.8**. This

modification had little effect on the pressure-flow curve as expected and since it is probably not a reasonable physiological adaptation we shall not consider it further.

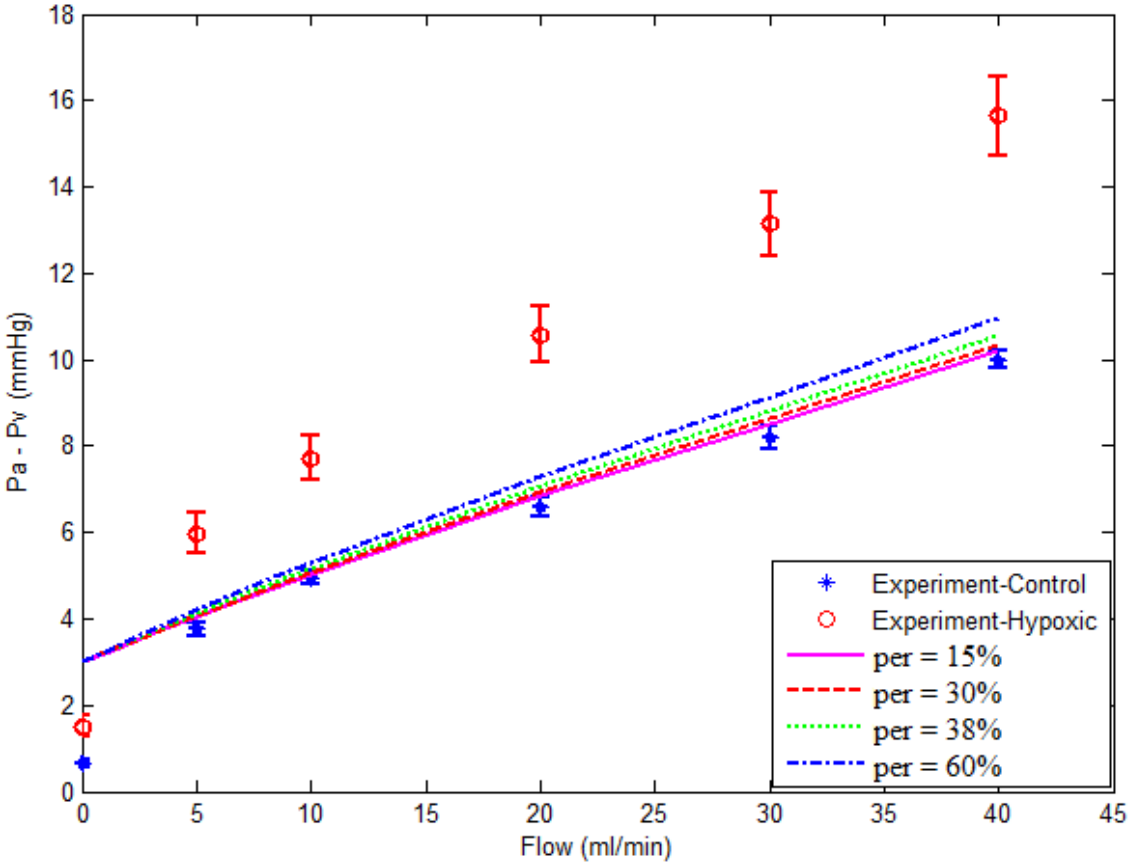


Figure 6.8: Model simulation result of difference between pulmonary arterial and venous pressure versus flow showing the effect of removing 15% - 60% of arteries greater than 200 micron diameter.

6.4.5 Combined effects of distensibility, capillary sheet area, and rarefaction

Finally we incorporated arterial and venous distensibility (D), reduction in capillary sheet area (*CSA*), and arterial rarefaction (*AR*) into the model and compared the resulting pressure - flow curves with the experimental chronic hypoxia experimental data. We used the previously reported experimental values, i.e., arterial distensibility = 1.3

%/mmHg, venous distensibility = 0.2 %/mmHg, capillary sheet area reduction = 15%, and arterial rarefaction = 38% (Hislop & Reid, 1976) (D *+CSA-15%+AR-38%*). The model pressure – flow curve and corresponding hypoxic data is shown in **Figure 6.9**. We observe that with these particular parameter settings, the model simulated pressure-flow curve approaches the hypoxia experimental data, but yet still falls short of capturing the substantial increase in pressure measured in lungs of rats exposed to chronic hypoxia. We also ran the simulation using arterial rarefaction of 55% which led to a substantial increase in the pressure-flow curve, but is well above any previously reported value in the literature.

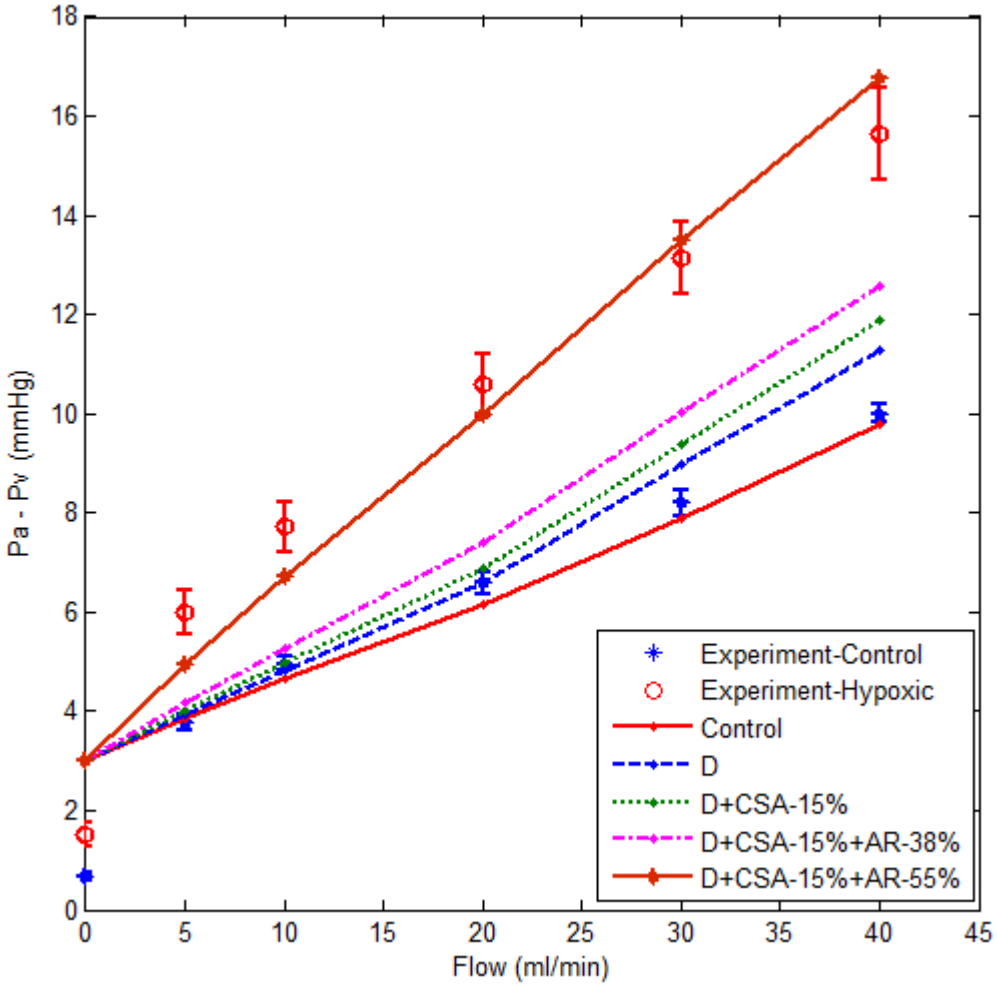


Figure 6.9: Model simulation of pressure versus flow showing the effect of decrease in distensibility alone (*D*), distensibility and capillary surface area -15% (D+*CSA*) and the combined effect of decrease in distensibility, capillary surface area and arterial rarefaction (38 and 55 %) (D+*CSA-15%+AR-38% and* D+*CSA-15%+AR-55%).*

The reduction in arterial and capillary volumes is reported to be a factor contributing to the observed increase in pressure with hypoxia (Molthen, Wietholt, Haworth, & Dawson, 2002). For further comparison, the calculated model volume distribution with arteries, veins and capillaries for each of the simulation described from 6.4.1 to 6.4.5 is also shown in **Table 6.4**.

Model Simulation	*Va*	*Vc*	*Vc*	*VL*
Control	0.235	0.584	0.225	1.045
D	0.213	0.584	0.252	1.05
CSA (15%)	0.239	0.507	0.225	0.972
D + *CSA (15%) + AR (38%)*	0.210	0.507	0.252	0.969
D + *CSA (15%) + AR (55%)*	0.199	0.507	0.252	0.958

Table 6.4: Model simulation results of volume and resistance within arteries, capillaries and veins at a flow of 30 ml/min. *Va*, *Vc* and *Vv* represent arterial, capillary and venous volume; *VL* is the total volume.

7. DISCUSSION AND CONCLUSION

The objective of this project was to adapt a previously developed model of the dog pulmonary circulation to a model appropriate for the rat. The project consisted of two major tasks. The first was to develop and test a user-friendly computational model. The second was to use the model to investigate changes in pulmonary hemodynamics and vessel morphometry that have been implicated as contributing factors to the increase pulmonary arterial pressure observed in the chronic hypoxia rat model of pulmonary hypertension.

Model Development and Validation

The rat model geometry for arteries and veins were established using a continuum model proposed by Krenz et al. (Krenz, 1992). The model is characterized by 18 orders of arteries and 19 orders of veins. The average distensibility (% increase in diameter over the undistended diameter) for the model arteries and veins is 2.8 %/mmHg and 1.6 %/mmHg, respectively. These arterial and venous trees are connected by a capillary sheet with an area of 0.123 cm^2. Using this morphomtery we showed that the arterial-capillary-venous resistances, volumes and compliances calculated from the model agreed well with experimental estimates.

With steady flow, the model can simulate the characteristic nonlinear shape of the mean pulmonary pressure–flow curve. The lung model could potentially be used to simulate the vasoactive response of vasoconstrictors or dilators. For the dynamic model, the lung model can be used to test the response of arterial/double occlusion at various times in the cardiac cycle. Also, the model can incorporate respiratory action by

following the inspiratory – expiratory air volume and the resulting geometric, biomechanic and rheologic effects. This large scale model can be used for interpreting experimental measurements and forming new testable hypotheses. The final version of the model software is available for MATLAB 7.14 (R2012a); it has an easy to use GUI suitable for new users and easy modification of the input parameters and output results. The speed of the program makes it easy to perform multiple steady state and dynamic simulations and has great potential in the evaluation and testing of hypotheses regarding the hemodynamic function of the rat pulmonary circulation.

Simulating Chronic Hypoxia

The model was used to simulate the effects of previously proposed alterations in the pulmonary vascular bed on observed changes in vascular pressure with chronic hypoxia as a model of pulmonary hypertension. The first step was to compare the control rat experimental results with model simulation output. The steady state parameter values used in the experiment and the model were the same except for airway pressure (*PA*) as shown on **Table 6.1**. In the experiment, *PA* of 6 mmHg was chosen because this high *PA* was required for the contrast agent to reach the lung for subsequent CT imaging. The drawback of using the high *PA* is that the lung may be transitioning from zone-1 (*PA* > *Pa* > *Pv*) to zone-2 (*Pa* > *PA* > *Pv*) as flow is increased. Since the model does not account for the hydrostatic pressure gradient due to gravitational effects, we used an airway pressure of 3 mmHg in our simulations. **Figure 6.2** shows good agreement between the model simulations and the experiment data.

We then studied the effect of vessel distensibility as shown in **Figure 6.3** and **Figure 6.4** (Molthen, Karau, & Dawson, 2004). Results indicated that as arterial

distensibility is decreased *Pa* increased, consistent with increased pressure in rigid arteries. In contrast, changing venous distensibility had less effect on *Pa* when compared to arterial distensibility, over the range of distensibility coefficients and flows studied. **Figure 6.5** shows that reducing both arterial and venous distensibility to values reported for hypoxic lungs was not alone sufficient to account for the measured increase in *Pa* in the hypoxic lungs.

We went on to consider reported reductions in arterial and capillary volumes as factors contributing to the observed increase in pressure with hypoxia. **Figure 6.6** shows that reducing capillary surface area by 15% resulted in only a modest increase in arterial pressure. However, there was a substantial increase in the arterial pressure when the capillary surface area was reduced by 45%. Reducing the number of small arteries (diameter < 200 μm) also had a substantial impact on *Pa* (**Figure 6.7**). This is consistent with the notion that reducing the number of parallel vascular pathways increases vascular resistance and pressure within the lung (Hislop & Reid, 1976).

The combined effect on *Pa* of reducing vessel distensibility, capillary surface area and the number of small arteries was simulated in an attempt to better fit the observed hypoxia pressure-flow data. From **Figure 6.9** it appears that with the reported values of distensibility, 15% reduction in capillary surface area, and 38% reduction in the number of arteries smaller than 200 microns, the model simulations do not fully account for the reported increase in *Pa* with hypoxia. For the model to give a better fit to the data at least 55% of the arteries under 200 microns would need to be removed, which is much larger than the 38% reported in the literature (Hislop & Reid, 1976).

Molthen et al. also reported a reduction in arterial volume in rats when exposed to chronic hypoxia (Molthen, Wietholt, Haworth, & Dawson, 2004). The volume distribution for arteries, veins, and capillaries for each model simulation for a flow of 30 ml/min is shown in **Table 6.4**. By changing the arterial and venous distensibility (D) to the reported hypoxic values, the arterial volume is reduced from 0.23 to 0.21 ml while the venous volume increased from 0.225 to 0.252 ml, with the total volume of 1.0 ml. The reduction in SA reduced the capillary volume from 0.584 ml to 0.507 ml, thereby reducing the total volume to 0.97ml. With the combination of arterial rarefaction of 38% decrease in arterial and venous distensibility, and 15% reduction in capillary surface area, the arterial volume dropped to 0.21 ml and venous volume increased to 0.25 ml to give a total volume of 0.96 ml.

In summary, using this large scale pulmonary circulation model, we tested the common structural hallmarks of pulmonary vascular remodeling including the decrease in arterial and venous distensibility, reduction in capillary surface area and reduction in the number of small arteries. We conclude that these factors are not alone sufficient to account for the reported increase in pulmonary arterial pressure in response to chronic hypoxia induced pulmonary hypertension. Thus, the critical open questions from our work are whether the model accurately incorporates the physiological phenomena previously described and whether there may be other factors contributing to the increase in *Pa* with hypoxia. As indicated throughout this work, we have based our simulations on previously proposed data and hypotheses. However, there is substantial conflicting data in the literature regarding mechanisms of increased *Pa*. Some findings suggest that pulmonary hypertension may not result from the structural loss of blood vessels nor from

the structural narrowing of the vessels alone. They reported an increase in total pulmonary vessel length, volume, endothelial surface area and number of endothelial cells *in vivo* (Howell, Preston, & McLoughlin, 2003). Arterial rarefaction has been reported in rodents exposed to chronic hypoxia (Hislop & Reid, 1976; Rabinovitch, Gamble, Nadas, Miettinen, & Reid, 1979; Jones & Reid, 1995), yet others found no such loss (Meyrick & Reid, 1979; Emery et al., 1981; Finlay et al.1986; Rabinovitch, Chesler, & Molthen, 2007). Moreover there is even evidence of new pulmonary vessel formation in response to chronic hypoxia (LaManna et al. 1992; Smith, 1997; Griffioen & Molema, 2000). Thus, we concur that hypoxia–induced remodeling of the pulmonary circulation is a complex process involving numerous interactive events (Stenmark, Fagan, & Frid, 2006).

Future Work

The present model has been validated under zone 3 conditions (Chapter - Model validation, **Table 5.3**. Since most of the lung is in zone 3, the next step would be to repeat the hypoxia experiment under zone 3 conditions and compare the results with the model simulation results. Also the model would need to be modified so that it can accurately simulate zone 1 and zone 2 conditions by incorporating the hydrostatic pressure gradient due to gravitational effects within the lung.

Another future task would be to test the dynamic model simulation results with the published dynamic data under different experimental conditions. Currently the dynamic model flow input options are rectified sine wave and sinusoidal pulses. The model could be expanded to incorporate experimentally measured flow data.

The current version of the model uses a homogeneous, dichotomous branching structure. However imaging of the pulmonary vasculature has revealed a complex heterogeneous branching pattern which could be incorporated into our model and may have the potential for considerable influence on the intravascular pressure distribution. The sheet flow model used to represent the capillary bed provides an average value of pressure throughout the entire capillary network. Again, it would be feasible to update the model to represent more realistic capillary network geometry as implemented by Burrowes et al. (Burrowes, Tawhai, & Hunter, 2004). This model generates a continuous network of capillaries over adjacent model alveoli in a single alveolar sac.

The model assumes the arterial and venous distensibility is diameter independent, which might not be reasonable. With our current method, the impact of changes in the precapillary venules that controls the resistance is not measurable. These small vessels are most likely more variable with remodeling. They are much muscularized, and hence could have more of an impact on the pressure. Another limitation of the model is that it does not allow for a change in vessel length as the diameter changes due to vasoconstriction. Such a coupled change in length and diameter is suggested in studies with acute vasoconstriction induced by lung treatment with serotonin (Wideman.Jr & Hamal, 2011; Martinez-Lemus, Hill, Bolz, Pohl, & Meininger, 2004).

BIBLIOGRAPHY

Albert, R. K., Lamm, W. J., Rickaby, D. A., & al-Tinawi, A. (1993). Lung inflation distends small arteries(<1mm) in excised dog lungs. *J.Appl.Physiol, 75*(6), 2595-2601.

Alessandro, R. (2005). Effect of perfusate temperature on pulmonary vascular resistance and compliance by arterial and venous occlusion in the rat. *European Journal of Applied Physiology, 93*(4), 435-439.

al-Tinawi, A., Krenz, G. S., Rickaby, D. A., Linehan, J. H., & Dawson, C. A. (1994). Influence of hypoxia and serotonin on small pulmonary vessels. *J.Appl.Physiol,76, 76*(1), 56-64.

Barer, G. R., Bee, D., & Wach, R. A. (1983). Contribution of Polycythemia to Pulmonary Hypertension in Simulated High Altitude in Rats. *J Physiol, 336*, 27-38.

Bassingthwaighte, J. B. (2000). Strategies for the Physiome Project. *Annals of Biomedical Engineering, 28*(8), 1043-1058.

Bennett, S. H., Goetzman, B. W., Milstein, J. M., & Pannu, J. S. (1996). Role of arterial design on pulse wave reflection in a fractal pulmonary network. *J Appl Physiol, 80*(3), 1033–1056.

Bolle, I., Eder, G., Takenaka, S., Ganguly, K., Karrasch, S., Zeller, C., . . . Schulz, H. (2008). Postnatal lung function in developing rat. *J Appl Physiol, 104*(4), 1167-1176.

Borg, T. J., & Hunter, P. J. (2003). Integration from proteins to organs: the Physiome Project. *Nature Reviews Molecular Cell Biology, 4*, 237-243.

Bshouty, Z., & Younes, M. (1990). Distensibilty and pressure flow relationship of the pulmonary circulation.I.Single-vessel model. *J.Appl.Physiol, 68*(4), 1501-1513.

Burrowes, K. S., & Tawhai, M. H. (2006). Computational predictions of pulmonary blood flow gradients: Gravity versus structure. *Respir. Physiol. Neurobiol, 154*(3), 515–523.

Burrowes, K. S., Hunter, P. J., & Tawhai, M. H. (2005). Anatomically-based finite element models of the human pulmonary arterial and venous trees including supernumerary vessels. *J Appl Physiol, 99*(2), 731–738.

Burrowes, K. S., Swan, A. J., Warren, N. J., & Tawhai, M. H. (2008). Towards a virtual lung: multiscale,multi-physics modelling of the pulmonary system. *Phil. Trans. R. Soc. A, 366*(1879), 3247-3263.

Burrowes, K. S., Tawhai, M. H., & Hunter, P. J. (2004). Modeling RBC and neutrophil distribution through an anatomically based pulmonary capillary network. *Ann Biomed Eng, 32*(4), 585–595.

Cahill, E., Rowan, S. C., Sands, M., Banahan, M., Ryan, D., Howell, K., & McLoughlin, P. (2012). The pathophysiological basis of chronic hypoxic pulmonary hypertension in the mouse: vasoconstrictor and structural mechanisms contribute equally. *Experimental Physiology*. doi:10.1113/expphysiol.2012.065474

Capro, J. D., Barry, B. E., Foscue, H. A., & Shelburne, J. (1980). Structural and biomedical changes in Rat Lungs Occuring During Exposure to Lethal and Adaptive Doses of Oxygen. *Am Rev Respir Dis, 122*, 123-143.

Clarke, S., Baumgardt, S., & Molthen, R. (2010). The effect of ACE inhibition on the pulmonary vasculature in a combined model of chronic hypoxia and pulmonary

arterial banding in Sprague Dawley rats. In J. B. Robert C. Molthen (Ed.), *Medical Imaging 2010: Biomedical Applications in Molecular, Structural, and Functional Imaging. 7626, 762614.* Bellingham WA: SPIE.

Clough, A. V., Audi, S. H., Molthen, R. C., & Krenz, G. S. (2006). Lung Circulation Modeling: Status and Prospects. *Proceedings of IEEE Special Issue on The Physiome and Beyond, 94*(4), 753-768.

Crapo, J. D., Barry, B. E., Foscue, H. A., & Shelburne, J. (1980). Structural and biomedical changes in Rat Lungs Occuring During Exposure to Lethal and Adaptive Doses of Oxygen. *Am Rev Respir Dis, 122*, 123-143.

Dawson, C. A., Krenz, G. S., Karau, K. L., Haworth, S. T., Hanger, C. C., & Linehan, J. H. (1999). Structure-function relationships in the pulmonary arterial tree. *J Appl Physiol, 86*(2), 569–583.

Dawson, C. A., Rickaby, D. A., & Linehan, J. H. (1986). Location and mechanisms of pulmonary vascular volume changes. *J.Appl.Physiol, 60*(2), 402-409.

Dhadwal, A., Wiggs, B., Doerschuk, C. M., & Kamm, R. D. (1997). Effects of anatomic variability on blood flow and pressure gradients in the pulmonary capillaries. *J. Appl. Physiol, 83*(5), 1711–1720.

Dhawal, A. S. (1993). A computational model for Pulmonary Microcapillary Blood Flow, Masters Thesis, Massachusetts Institute of Technology.

Fry, D. L., Thomas, L. J., & Greenfeild, J. C. (1980). Flow in collapsible tubes. In D. J. Patel, & R. N. Vaishnav, *Basic Hemodynamic and its Role in Disease Processes* (pp. 407-424). Baltimore: University Park Press.

Fung, Y. C. (1997). *Biomechanics: Circulation* (2nd ed.). New York: Springer.

Fung, Y., & Sobin, S. (1972). Elasticity of the Pulmonary Alveolar Sheet. *Circ Res. 30*, 451-469.

Fung, Y., & Sobin, S. (1969). Theory of sheet flow in lung alveoli. *J. Appl. Physiol. 26(4)*, 472-478.

G.S.Krenz, J. (1992). A fractual continuum model of pulmonary arterial tree. *Journal of Applied Physiology*.

Gan, R. Z., & Yen, R. T. (1994). Vascular impedance analysis in dog lung with detailed morphometric and elasticity data. *J.Appl.Physiol, 77*(2), 706-717.

Gan, R. Z., Tian, Y., Yen, R. T., & Kassab, G. S. (1993). Morphometry of the dog pulmonary venous tree. *J.Appl.Physiol, 75*(1), 432-440.

Gao , Y., & Usha Raj, J. (2004). Role of veins in regulation of pulmonary circulation. *Am J Physiol, 288*(2), 213-226.

George, R. B., Light, R. W., Matthay, M. A., & Matthay, R. A. (2005). *Chest medicine - Essentials of Pulmonary and Critical Care Medicine.* Philadelphia: Lippincott Williams & Wilkins.

Glazier, J. B., Hughes, J. M., Maloney, J. E., & West, J. B. (1969). Measurements of capillary dimensions and blood volume in rapidly frozen lungs. *J. Appl. Physiol, 26*(1), 65-76.

Guyton, A. C., & Hall, J. E. (1996). *Human physiology and mechanisms of disease* (6 ed.). Philadelphia: Saunders.

Gwenda R. Barber, D. B. (1982). Contribution of Polycythaemia to Pulmonary Hypertension in Simulated High Altitude in Rats. *Journal of Physiology*.

Haddy, F. J. (1960). Effect of histamine on small and large presseures in the dog foreleg. *Am J Physiol, 198*(1), 161-168.

Haworth, S. T. (1996). Mathematical Model of the Pulmonary Circulation: Effect of Lung Inflation and Cardiac Output. PhD Dissertation, Marquette University.

Haworth, S. T., Linehan, J. H., Bronikowski, T. A., & Dawson, C. A. (1991). A hemodynamic model representation of the dog lung. *J Appl Physiol, 70*, 15-26.

Hillyard, R., Anderson, J., & JU Raj . (1991). Segmental vascular resistance in isolated perfused rat lungs. Influence of vasomotor tone and cyclooxygenase and lipooxygenase inhibition. *Circulation Research, 68*, 1020-1026.

Hislop A, & Reid L. (1976, October). New findings in pulmonary arteries of rats with hypoxia-induced pulmonary hypertension. *Br J Exp Pathol, 57*(5), 542-554.

Hope D.Intengan, G. T.-S. (1999). Resistance Artery Mechanics,Structure and Extracellular Components in Spontaneously Hypertensive Rats. *Journal of the American Heart Association*.

Hopkins, N., & McLoughlin, P. (2002). The structural basis of pulmonary hypertension in chronic lung disease:remodelling,rarefaction or angiogenesis? *J Anat, 201*(4), 335-348.

Horsfield, K. (1978). Morphometry of the small pulmonary arteries in man. *Circ. Res*, 593-597.

Horsfield, K., & Gordon, W. (1891). Morphometry of pulmonary veins in man. *Lung, 159*(1), 211-218.

Howell, K., Preston, R. J., & McLoughlin, P. (2003). Chronic hypoxia causes angiogenesis in addition to remodelling in the adult rat pulmonary circulation. *J.Appl.Physiol, 547(P t1)*, 133-145.

Huang, Y., Doerschuk, C. M., & Kamm, R. D. (2001). Computational modeling of RBC and neutrophil transit through the pulmonary capillaries. *J Appl Physiol, 90*(2), 545–564.

Hunter , P. J., & Borg, T. K. (2003). Integration from proteins to organs: the Physiome Project. *Nature Reviews Molecular Cell Biology, 4*, 237-243.

Hunter, P. J., Crampin, E. J., & Nielsen, P. (2008). Bioinformatics, multiscale modeling and the IUPS Physiome Project. *Briefings in Bioinformatics, 9*(4), 333-343.

Hyduk, A., Croft, J., Ayala, C., Zheng, K., Zheng, Z., & Mensah, G. (2005). Pulmonary Hypertension Surveillance- United States,1980-2002. *MMWR;54*, 1-28.

Intengan, D. H., Thibault, G., Li, J.-S., & Schiffrin, E. L. (1999). Resistance Artery Mechanics, Structure, and Extracellular Components in Spontaneously Hypertensive Rats : Effects of Angiotensin Receptor Antagonism and Converting Enzyme Inhibition. *Circulation*, 2267-2275.

Iqbal, A. (2005). *Mananatomy*. Retrieved 2011, from http://www.mananatomy.com/wp-content/uploads/2011/04/pulmonary_circulation.jpg

J.H.Weiner. (1974). *Journal of applied physiology*.

Jiang, Z. L., Kassab, G. S., & Fung, Y. C. (1994). Diameter-defined Strahler system and connectivity matrix of the pulmonary arterial tree. *J Appl Physiol, 76*(2), 882-892.

Jin, Y., Calvert, T. J., Chicoine, L. G., & Nelin, L. D. (2009). Chronic hypoxia decreases arterial compliance in rat lungs. *FASEB J, 619*(23).

Jones, R., & Reid, L. (1995). Vascular remodelling in the clinical and experimental pulmonary hypertensions. In J. Bishop, G. Laurent, & J. Reeves, *Pulmonary Vascular Remodelling.* Portland Press Ltd.

Kent E.Pinkerton, P. G. (1992). *Treatise on pulmonary toxicology: comparative biology of the normal lung. Boca Raton (FL): CRC Press; 1992;p. 121-135* (In: Parent RA editors ed.)

Kiani, M., & Hudetz, A. (1991). A semi-empirical model of apparant blood viscosity as a function of vessel diameter and discharge hematocrit. *Biorheology, 28*(1-2), 65-73.

Krenz, G. S., Linehan, J. H., & Dawson, C. A. (1992). A Fractal Continuum Model of the Pulmonary Arterial Tree. *J Appl Physiol, 72*(6), 2225-2237.

Krenz, S. G., & Dawson, C. A. (2003). Flow and pressure distributions in vascular tress consisting of distensible vessels. *Am J Physiol, 284*(6), 2192-2203.

Krishnan, A., Linehan, J. H., Rickaby, D. A., & Dawson, C. A. (1986). Cat lung hemodynamics: comparison of experimental results and model predictions. *J Appl Physiol, 61*(6), 2023-2034.

Leung, M., Dumont, G. A., Sandor, C., & Potts, J. (2006). Estimating arterial stiffness using transmission line model. *28th IEEE Engineering in Medicine and Biology Conference.* Newyork.

Levitzky, M. G. (2006). Teaching the effects of gravity and intravascularalveolar pressures on the distribution of pulmonary blood flow using a classic paper by West et al. *Advan in Physiol Edu*(30), 5-8.

Lin, C. L., Tawhai, M. H., McLennan, G., & Hoffmann, E. A. (2007). Characteristics of the turbulent laryngeal jet and its effect on airflow in the human intra-thoracic airways. *Respir Physiol Neurobiol, 157*(2-3), 295–309.

Linehan, J. H., Dawson, C. A., Rickaby, D. A., & Bronikowski, T. A. (1986). Pulmonary vascular compliance and viscoelasticity. *J Appl Physiol, 61*(5).

Linehan, J. H., F.deMora, Bronikowski, T. A., & Dawson, C. A. (1988). Hemodynamic modelling of vascular occlusion experiments in cat lung. *Advances in Bioengineering, 8*, 139-142.

Linehan, J. H., Haworth, S. T., Nelin, L. D., Krenz, G. S., & Dawson, C. A. (1992). A simple distensible vessel model for interpreting pulmonary vascular pressure-flow curves. *J Appl Physiol, 73*(3), 987-994.

Martinez-Lemus, L., Hill, M., Bolz, S., Pohl, U., & Meininger, G. (2004). Acute mechanoadaptation of vascular smooth muscle cells in response to continuous arteriolar vasoconstriction: implications for functional remodeling. *FASEB J, 18*(6), 708-710.

Mazzone, R. (1980). Influence of vascular and trnaspulmonary pressure on the functional morphology of the pulmonary circulation. *Microvasc.res,20,* 295-306.

Miller, W. (1893). The structure of the lung. *J.Morphology,* 165-182.

Milnor, W. R. (1989). *Hemodynamics, 2nd ed.* Baltimore: Williams & Wilkins.

Molthen, C. R., Haworth, S. T., Gordon, J. B., Krenz, G. S., & Clough, A. V. (2005). Chronic hypoxia effects active tone differently in the pulmonary arteries of Brown-Norway, Sprague-Dawley, and Fawn-Hooded rats. *FASEB.*

Molthen, R. C., Gordon, J. B., Krenz, G. S., & Clough, A. V. (2007). Effect of Chronic Hypoxia on Rat Pulmonary Venous Tree,Proceedings of the American Thoracic Society (PATS), Abstracts Issue, May 18-23.

Molthen, R. C., Karau, K. L., & Dawson, C. A. (2004). Quantitative models of the rat pulmonary arterial tree morphometry applied to hypoxia-induced arterial remodeling. *J Appl Physiol, 97*(6), 2372-2384.

Molthen, R., Heinrich, A., Haworth, S., Krenz, G., & Gordon, J. (2004). The effect of captopril treatment on chronic hypoxia induced pulmonary vasccular remodeling in the Fawn-Hooded, Sprague-Dawley, and Brown-Norway rat. *FASEB*.

Molthen, Wietholt, C., Haworth, S., & Dawson, C. (2002). Estimation of Pulmonary Arterial Volume Changes in the Normal and Hypertensive Fawn-Hooded Rat from 3D Micro-CT Data. In C. C. Clough (Ed.), *Physiology and Function : Methods, Systems, and Applications* (pp. 266-275). SPIE 4683.

Molthen, Wietholt, C., Haworth, S., & Dawson, C. (2004). Estimation of Pulmonary Arterial Volume Changes in the Normal and Hypertensive Fawn- Hooded Rat from 3D MocroCT-Data. *SPIEE*.

Morell, N. W., & Hughes, J. (2001). *Pulmonary Circulation: From Basic Mechanisms to Clinical Practice.* Imperial College Press.

Noordergraaf, A. (1969). *Hemodynamics. In Biological Engineering.* (H. P. Schwan, Ed.) New York: McGraw-Hill.

Palladino, J., Drzewiecki, G., & Noordergraaf, A. (2000). *Modeling Strategies in Physiology.* CRC Press LLC.

Parker, J. C., Cave, C. B., Ardell, J. L., Hamm, C. R., & Williams, S. G. (1997). Vascular tree structure affects lung blood flow heterogeneity simulated in three dimensions. *J Appl Physiol, 83*(4), 1370–1382.

Parker, J. C., Gillespie, M. N., Taylor, A. E., & Martin, S. L. (1999). Capillary filteration coefficient,vascular resistance and compliance in isolated mouse lungs. *J Appl Physiol, 87*(4), 1421-1427.

PeterJ.Hunter, E. J. (2008). Bioinformatics,Multiscale modeling and IUPS Physiome Project. *Oxford Journals*, 333-343.

Presson, R. G. (1998). Anatomic distribution of pulmonary vascular compliance. *J Appl Physiol, 84*, 303-310.

Presson, R. G., Audi, S. H., Hanger, C. C., Zenk, G. M., Sidner, R. A., Linehan, J. H., . . . Dawson, C. A. (1998). Anatomic distribution of pulmonary vascular compliance. *J Appl Physiol, 84*(1), 303-310.

Rabinovitch, M., Chesler, N., & Molthen, R. C. (2007). Point:Counterpoint: Chronic hypoxia-induced pulmonary hypertension does/does not lead to loss of pulmonary vasculature. *J Appl Physiol, 103*, 1447-1451.

Rabinovitch , M., Gamble , W., Nadas, A., Miettinen, O., & Reid, L. (1979). Rat pulmonary circulation after chronic hypoxia: hemodynamic and structural features. *Am J Physiol, 236*(6), 18-27.

Ramakrishna, M. (2009). Capillary Perfusion Kinematics in Lungs of Oxygen-Tolerant Rats.Master's Thesis, Marquette University.

Rouge wave software. (n.d.). Retrieved from http://www.roguewave.com/products/imsl-numerical-libraries.aspx

Sasaki, S., Yasuda, K., McCully, J. D., & LoCicero, J. (1997). Development of an isolated, pulsatile blood-perfused rat lung model for evaluating the preserved lung functions. *Surgery Today, 27*(12), 1154-1159.

Shifren, A., Durmowicz, A. G., Knusten, R. H., Hirano, E., & Mecham, R. P. (2007). Elastin protein levels are a vital modifier affecting normal lung development and susceptibility to emphysema. *Am J Physiol Lung Cell Mol Physiol*, 778-787.

Shingrani , R., Krenz, G., & Molthen, R. (2010). Automation process for morphometric analysis of volumetric CT data from pulmonary vasculature in rats. *Comput.Methods Programs Biomed, 1*(97), 62-77.

Singhal, S., Henderson, R., Horsfeild, K., Harding, K., & Cumming, G. (1973). Morphometry of the human pulmonary arterial tree. *Circ Res, 33*, 190-197.

Smith, J. C., & Mitzner, W. (1980). Analysis of pulmonary vascular interdependence in excised dog lobes. *J Appl Physiol, 48*(3), 450-467.

Stenmark, K. R., Fagan, K. A., & Frid, M. G. (2006). Hypoxia-Induced Pulmonary Vascular Remodelling. *Circulation Research, 99*, 675-691.

Steven T. Haworth, J. H. (1991). A hemodynamic model representation of the dog lung. *Journal of Applied Physiology*.

Suki, B., Alencar, A. M., Frey, U., Ivanov, P. C., Buldyrev, S. V., Majumdar, A., . . . Mishima, M. (2003). Fluctuations, noise and scaling in the cardio-pulmonary system. *Fluctuations and Noise Lett, 3*, R1- R25.

Tawhai, M. H., Clark, A. R., & Burrowes, K. S. (2011). Computational models of pulmonary circulation: Insights and move towards clinically directed studies. *Pulm Circ, 1*(2), 224-238.

Tawhai, M. H., Clark, A. R., & Burrowes, K. S. (2011). Computational models of pulmonary circulation: Insights and the move towards clinically directed studies. *Pulm Circ, 1*(2), 224-238.

Tawhai, M. H., Hoffman, E. A., & Lin, C. -L. (2009). The Lung Physiome: merging imaging- based measures with predictive computational models of structure and function. *Systems Biology and Medicine, 1*(1), 61-72.

Tawhai, M. H., Hunter, P., Tschirren, J., Reinhardt, J., McLennan, G., & Hoffman, E. A. (2004). CT-based geometry analysis and finite element models of the human and ovine bronchial tree. *J Appl Physiol, 97*(6), 2310-2321.

Tawhai, M. H., Nash, M. P., & Hoffman, E. A. (2006). An imaging-based computational approach to model ventilation distribution and soft tissue deformation in the ovine lung. *Acad. Radiol. 13*, 113–120.

Vanderpool, R. R., Kim, A. R., Molthen, R. C., & Chesler, N. C. (2010). Effects of acute Rho kinase inhibition on chronic hypoxia-induced changes in proximal and distal pulmonary arterial structure and function. *J Appl Physiol, 110*(1), 188-198.

Weibel, E. (1970). Morphometric Estimation of Pulmonary Diffusion Capacity,I.Model and Method. *Respir Physiol,11*, 54-75.

West, J. B., Dollery, C. T., & Naimark, A. (1964). Distribution of blood flow in isolated lung; relation to vascular and alveolar pressures. *J Appl Physiol, 8*(10), 713-724.

White, F. M. (1999). *Fluid Mechanics, 4th ed.* New York: McGraw-Hill.

Wideman.Jr, R. F., & Hamal, K. R. (2011). Idiopathic Pulmonary Arterial Hypertension: An Avian Model for Plexogenic Arteriopathy and Serotonergic Vasoconstriction. *J Pharmacol Toxiol Methods, 63*(3), 283-295.

Yen, R. T., Fung, Y. C., & Bingham, N. (1980). Elasticity of small pulmonary arteries in the cat. *J Biomed Eng, 102*(2), 170-177.

Yen, R. T., Zhuang, F. Y., Fung, Y. C., Ho, H. H., Tremor, H., & Sobin, S. S. (1983). Morphometry of cat pulmonary venous tree. *J Appl Physiol, 55*(1), 236-242.

Yin, Y., Choi, J., Hoffman, E. A., Tawhai, M. H., & Lin, C. L. (2010). Simulation of pulmonary air flow with a subject-specific boundary condition. *J Biomech, 43*(11), 2159-2163.

Zhao, L. (2011). Chronic Hyypoxia induced pulmonary hypertension in rat. *Drug Discovery Today: Disease Models, 7*(1-3), 83-88.

Zhuang, F. Y., Fung, Y. C., & Yen, R. T. (1983). Analysis of blood flow in cat's lung with detailed anatomical and elasticity data. *J.Appl.Physiol, 55*, 1341-1348.

GLOSSARY

A	capillary sheet area
A_{z2}	capillary sheet area in zone 2
A_{z3}	capillary sheet area in zone 3
a_1	intercept from linear regression of log N versus log D
a_2	intercept from linear regression of log ℓ versus log D
B	branching ratio
C	vascular compliance
Ca	segmental arterial volume
C_{b_j}	non viscoelastic compliance of order j
Cc	segmental arterial volume
Ccum	cumulative compliance
CL	total compliance
CSA	capillary sheet surface area
cv	coefficient of variation
C_{w_j}	viscoelastic compliance of order j
D	diameter of the vessels
D	arterial and venous distensibility
$D(0)$	unstressed vessel diameter at $P_{tm} = 0$
D_j	vessel diameter of j-th order

$Dterm$	terminal diameter
$D_{\min}$	minimum diameter of a single blood cell in a vessel
D_{MAX}	diameter of arteries/vein at large P_{tm}
$D(P_{tm})$	diameter of artery or vein for a specified P_{tm}
f	geometric friction factor
fh	frequency of heart beat
$h,\ h_c$	capillary sheet thickness
$h(0,0)$	capillary sheet thickness when P_{tm} and $P_{tp} = 0$
$h(P_{tm}, P_{tp})$	capillary sheet thickness for a specific P_{tm} and P_{tp}
Hct_f	feed hematocrit
Hct	blood hematocrit
h_{MAX}	maximum capillary sheet thickness
hs	minimum capillary sheet thickness
j	order number
k	parameter estimate
ℓ	length of the vessel
ℓ_j	vessel length of jth order
l_{cz2}	capillary sheet length in zone 2
l_{cz3}	capillary sheet length in zone 3
ℓ_{ref}	vessel length at V_{ref}
ℓ_{cref}	capillary sheet length at reference volume V_{ref}

l_c , ℓ_c	capillary sheet length
L	vascular inertance
Lcum	cumulative inertance
n	total number of artery or vein orders
n_p	number of posts
N	number of vessels
P	intravascular pressure
P*	estimate of mean capillary pressure
P_a	arterial pressure
Pa	mean pulmonary arterial pressure
PA	airway pressure
Pa_i	inlet arterial pressure
Pc	mean pulmonary capillary pressure
Pnode	pressure at the nodes
Ppl	pleural pressure
Pv, P_v	pulmonary venous pressure
Pv_o	outlet venous pressure
p_{wall}	pressure of the fluid at the tube wall
P_{tm}	transmural pressure
P_{tp}	transpulmonary pressure
$\ddot{P}_x$	perivascular pressure

P_{pl}	pleural pressure
P_{ext}	extravascular pressure
P_{outlet}	outlet pressure of an artery, vein or capillary sheet
P_{inlet}	inlet pressure of an artery, vein or capillary sheet
P_{out}	venous pressure
P_{w_j}	viscoelastic pressure at a particular order *j*
Q	blood flow
$\overline{Q}$	mean blood flow rate
R	Vascular resistance
R_0	vessel wall radius
r	ratio of successive vessel diameters
$R(j)$	cumulative arterial/venous vascular resistance from the inlet artery or outlet vein to the order *j*
Ra, R_a	segmental arterial resistance
Rc	segmental capillary resistance
Rcum	cumulative resistance
R_L	lobar resistance
RL	total resistance
R_v, *Rv*	segmental venous resistance
R_w	viscoelastic resistance
s/d	systolic diastolic ratio
t	time duration of the blood flow

V	lung volume (including tissue and trapped air)
Va	segmental arterial volume
Vc	segmental capillary volume
Vcum	cumulative volume
V_c	capillary sheet volume
$V(j)$	cumulative arterial/venous vascular volume from the inlet artery or outlet vein to the order j
V_L	lobar volume
VL	total volume
$VSTR$	vascular space to tissue ratio
V_{ref}	deflated lung volume plus trapped air and tissue volume
Vv	segmental venous volume
w, w_c	capillary sheet width
wc_{ref}	capillary sheet width at reference volume V_{ref}
w_j	viscoelastic parameter at a particular order j
$\overline{X}$	mean difference between experiment and model values

Greek Symbols

α_A	arterial distensibility
α_V	venous distensibility
α	distensibility coefficient
β_1	mean slope log N versus log D approximated by linear regression

β_2	mean slope log N versus log ℓ approximated by linear regression
γ	$D_{MAX} / D(0)$
μ	fluid viscosity
μ_a	apparent viscosity
μ_p	apparent viscosity of plasma
μ_c	blood viscosity in large vessels(>300 microns)
εc_{ref}	capillary sheet post diameter at V_{ref}
ε , εc	diameter of intermittent posts
δ	marginal plasma thickness
σ	standard deviation about the mean

APPENDIX 1

a. Input rat morphometry data file

N	*D*(cm)	*L*(cm)	*μ*(cP)	α(%/mmHg)	ϒ(ratio)
1	0.233598	0.577556	4.106547	2.8	2
2	0.177034	0.437705	4.100627	2.8	2
4	0.134167	0.331719	4.092859	2.8	2
8	0.101679	0.251396	4.082681	2.8	2
16	0.077058	0.190522	4.069378	2.8	2
32	0.058399	0.144389	4.052041	2.8	2
64	0.044258	0.109426	4.029534	2.8	2
128	0.033542	0.08293	4.00046	2.8	2
256	0.02542	0.062849	3.963147	2.8	2
512	0.019265	0.047631	3.915656	2.8	2
1024	0.0146	0.036097	3.855849	2.8	2
2048	0.011065	0.027357	3.781536	2.8	2
4096	0.008385	0.020732	3.690728	2.8	2
8192	0.006355	0.015712	3.582015	2.8	2
16384	0.004816	0.011908	3.455048	2.8	2
32768	0.00365	0.009024	3.311265	2.8	2
65536	0.002766	0.006839	3.156294	2.8	2
131072	0.002096	0.005183	3.007122	2.8	2
0	0.000562	0.01	1.92	2.6	2.066
262144	0.001896	0.004419	2.960961	1.6	1.5
131072	0.002501	0.005831	3.100003	1.6	1.5
65536	0.0033	0.007694	3.255754	1.6	1.5
32768	0.004355	0.010153	3.404633	1.6	1.5
16384	0.005746	0.013396	3.538004	1.6	1.5
8192	0.007582	0.017677	3.653377	1.6	1.5
4096	0.010005	0.023325	3.750566	1.6	1.5
2048	0.013202	0.030777	3.830659	1.6	1.5
1024	0.01742	0.040611	3.895483	1.6	1.5
512	0.022986	0.053586	3.947191	1.6	1.5
256	0.03033	0.070707	3.987963	1.6	1.5
128	0.04002	0.093299	4.01982	1.6	1.5
64	0.052807	0.123108	4.044535	1.6	1.5
32	0.06968	0.162442	4.063605	1.6	1.5

16	0.091943	0.214344	4.078256	1.6	1.5
8	0.121319	0.282828	4.089476	1.6	1.5
4	0.160082	0.373194	4.098047	1.6	1.5
2	0.211229	0.492433	4.104582	1.6	1.5
1	0.278718	0.649769	4.109557	1.6	1.5

The table shows the input morphometry file for the arteries, capillary and veins. In the above table, N is the number of vessels in each order; D is the diameter of each vessel when the transmural pressure is zero; μ is the blood viscosity in each order; α is vessel distensibility constant. The first 18 rows show the values for 18 orders of arteries. The capillary sheet values are given in row 19, where the value of D is the capillary sheet height. The next 19 rows show the values for 19 orders of veins.

b. List of functional model constants used in the rat model

i) Capillary sheet distension

Capillary sheet height vs. transmural pressure	Linear	Nonlinear
k_1(cm-H_20^{-1})	0	-0.11
h(0,∞) (cm)	0	1.42/1e4
h(0,0)(cm)	4.287/1e4	3.53/1e4
α(1/mmHg)	0.214/1e4	0.123
ϒ (ratio)	2	3.29

ii) Perivascular pressure

Bshouty_Smithmitzner (Arteries)
[k_1, k_2, k_3, k_4, k_5, k_6,] = [1.577e1, 4.20e-1, -1.495e1, -7.24e-2, -2.03e-3, -1.14e-3]

Bshouty_Smithmitzner (Veins)
[k_1, k_2, k_3, k_4, k_5, k_6,] = [3.51e1, -7.18e-1, -3.44e1, 3.99e-1, -4.12e-4, -8.10e-4]

Bshouty_Laifook (Arteries)
[k_1, k_2, k_3, k_4, k_5, k_6,] = [6.109e0, -7.363e-1, -7.014e0, 1.282e-1, 1.02e-2, -1.87e-3]

Bshouty_Laifook (Veins)
[k_1, k_2, k_3, k_4, k_5, k_6,] = [3.63e0, -4.34e-1, -3.01e0, -2.06e-1, -3.69e-3, -4.76e-3]

Haworth_Smithmitzner (Arteries)
[k_1, k_2, k_3, k_4] = [3.287e0, 6.396e0, -1.037e-2, 4.322e-1]

Haworth_Smithmitzner (Veins)
[k_1, k_2, k_3, k_4] = [3.24e0, 8.42e0, -1.275e-2, 3.109e-1]

Haworth_Laifook (Arteries)
[k_1, k_2, k_3, k_4] = [3.287, 6.396, -0.01037, 0.4322]

Haworth_Laifook (Arteries)
[k_1, k_2, k_3, k_4] = [3.24, 8.42, -.01275, .3109]

iii) Lung Air Volume

k_1 = 0.11
k_2 = 0.1471
Vm = 8.79 + 1
Ptp_FRC = 3.5

iv) Length vs. Volume (Smithmitzner model)

k_1 = 0.78
k_2 = -2.96

v) Capillary post diameter vs. Volume (Gilbased model)

k_1 = 0.96
k_2= 3.67

vi) Capillary sheet width/length vs. Volume (Smithmitzner model)

k_1 = 0.97
k_2 = 3.30

vii) Capillary sheet zone 2 conditions

Finite minimum thickness_hs = 1.417 / 10e4

viii) Viscosity Model

Kianihudetz model:

μ_p_p = 1.7

μ_c _ k_1 = 0.48

μ_c _ k_2 = 2.35

δ _ k_1 = 2.03

δ _ k_2 = 1.0

D_{min} = 2.7

Linehan model:

μ_p_p = 1.7

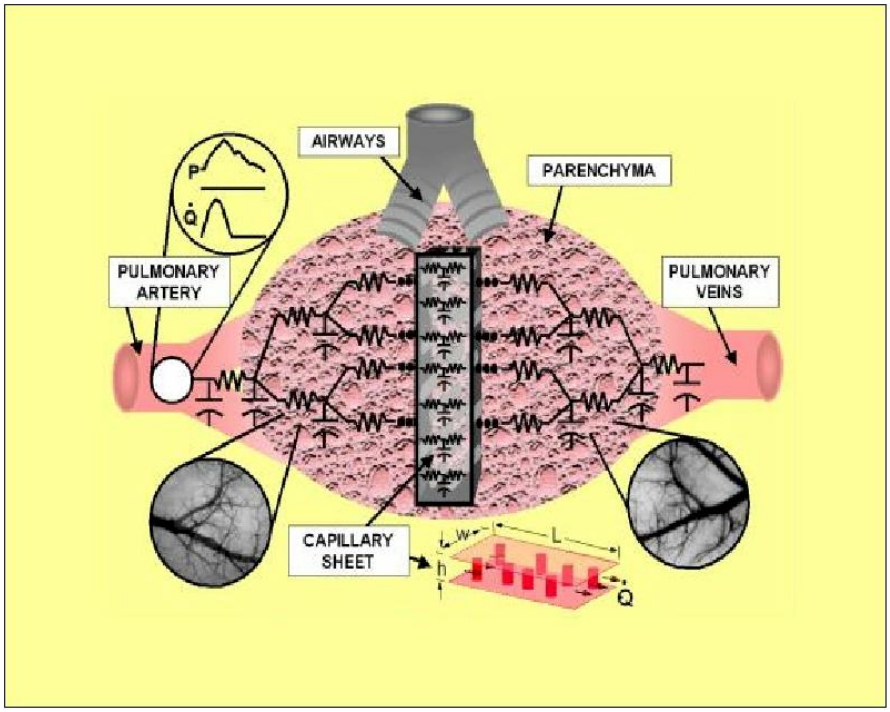
μ_c _ k_1 = 1.97
HctD_ k_1 = 2.90
HctD_ k_2 = 0.28
HctD_ k_3 = 12.04
HctD_ k_4 = 1

ix) Geometric friction factor

k_1 = -2.28
k_2 = 3.18
k_3 = -3.58
k_4 = 3.65

c. Instruction Manual

PC Physiome 2.0 Instruction Manual



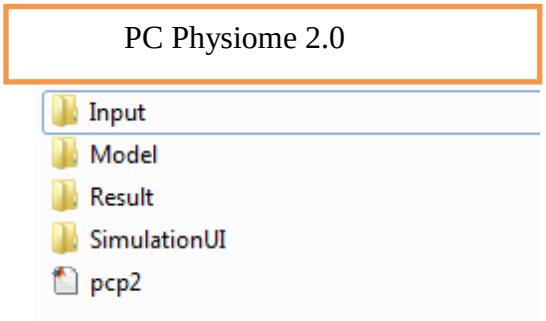
[Source: Steven T. Haworth]

LOOLU RAFEEQ, B.E
Marquette University

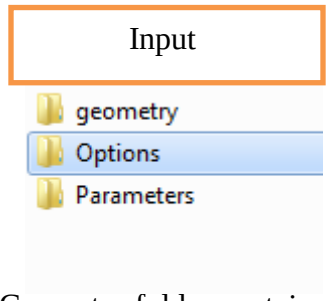
Prerequisites for running PCPhysiome_2.0

i. Matlab version 7.14 (R2012a) or higher. An older version for the software (R2007b) is also available.
ii. Operating system (Windows XP).
iii. MS Excel Installed (MS Office).
iv. Refer Chapter 5 in the thesis document for the details on the Software Overview and Design.

PCPhysiome_2.0 Directory Structure and contents

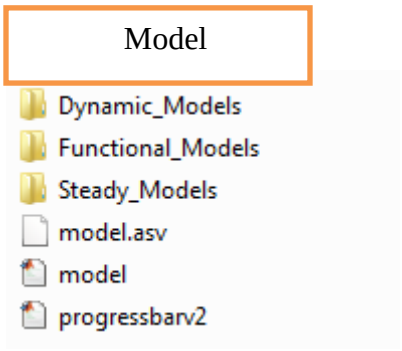


Folder Input has 3 folders



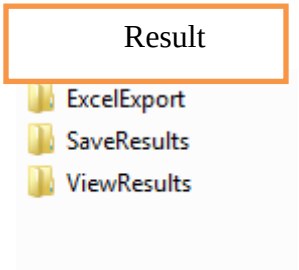
Geometry folder contains Morphometry files (Rat, dog and mouse), GUI for changing distensibility and number of arterial vessels.
Options folder contains the Options UI.
Parameters folder contains the constants used in the model, model parameter structure which loads the default model parameters and options. A multiple input file with 6 simulations is stored in the multiple input folders in the Parameters folder.

Folder Model has 3 folders and two files



Dynamic_Models folder contains the files for running the dynamic model.
Functional_Models folder contains the all the functional model files (Refer Chapter 5 in the thesis document for the list of functional models).
Steady_Models folder contains files for running the steady model.
'progressbarv2.m' shows the progress of the simulation and 'model.m' is the main program that is called when running the simulation.

Folder Result has 3 folders



Excel Export folder contains the code to write the results to a selected excel sheet.
Save Results folder contains code to save the results as matfiles to a selected location.
View result folder has UI file to view the Viewresults GUI. Simulation UI folder contains the GUI for the main Simulation User Interface and pcp2 is the main program to start PCPhysiome_2.0.

How to run the program

1) Start Matlab version 7.14 (R2012a)
2) Set C:\PCPhysiome_2.0 for Matlab's Current Directory.
3) In the Matlab command prompt type: 'clc' and 'clear all'.
4) Type **pcp2**.

⇨ Verify that the Main Simulation UI is displayed as shown in **Figure 1.1**.

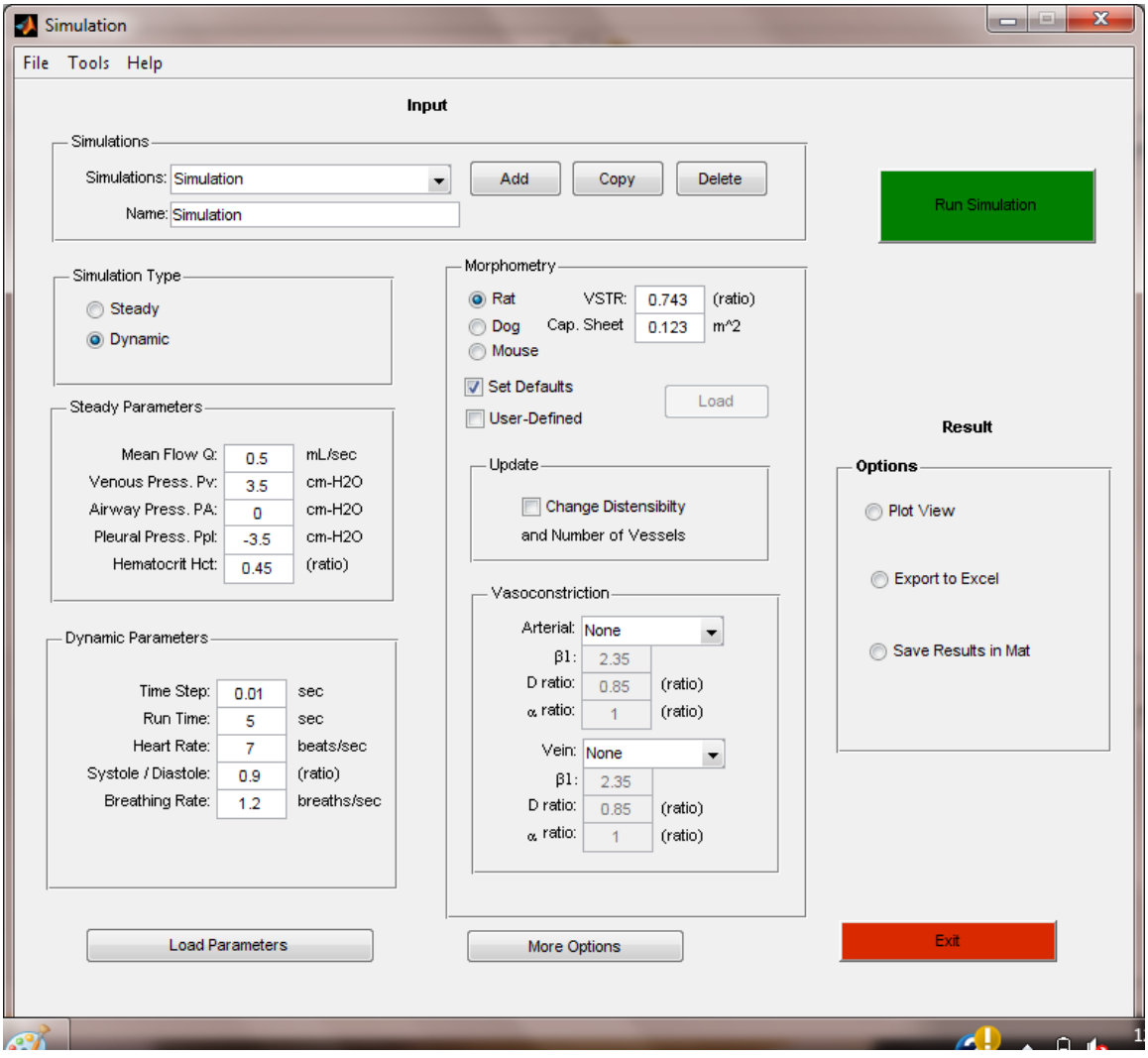


Figure 1.1: Screenshot of the Main Simulation User Interface.

Single Simulation:

1) Select a Simulation name : Eg: test1
2) Select a simulation type : Steady state (SS) or Dynamic (DY) -Choose the radio button
3) Select SS and DY parameters: Default parameters for the rat model are displayed in the steady and dynamic parameter edit boxes.
4) Select Morphometry: Default morphometry is the 'rat6.openfile' file. User can manually load other .text or .openfiles using the load button.
5) Select Functional Models and Options: Click 'More Options' button.
   ⇨ The options UI will be displayed as shown in the **Figure 1.2**. Default options and parameters will be displayed. These values can be set for all the simulations by changing values in 'ModelParams.m' file located at 'C:\PCPhysiome_2.0\Input\Parameters\ModelParams.m'. The options can be changed manually.

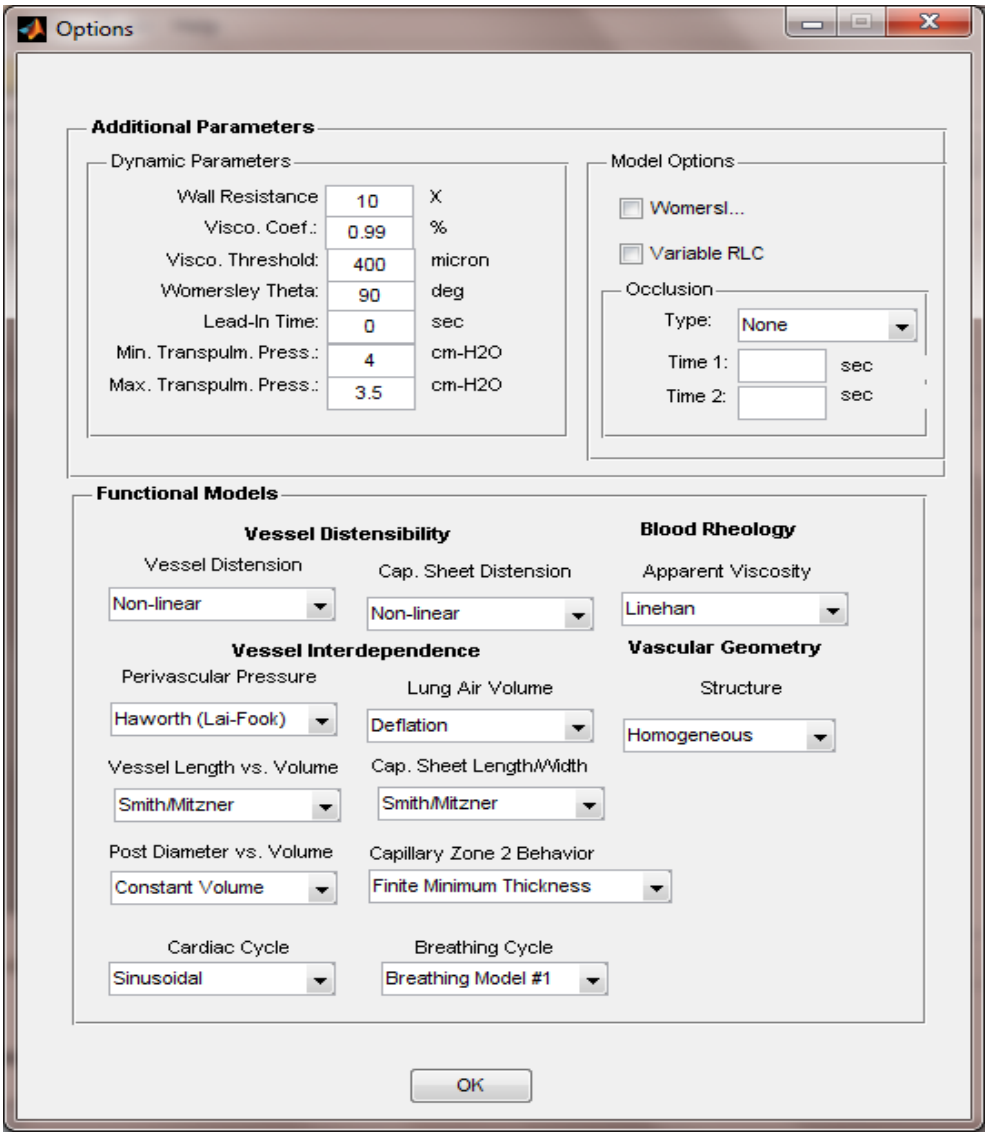


Figure 1.2: Screenshot of the Options User Interface.

Run the Simulation: Click on the green 'Run Simulation' button in the Simulation UI. A Progress bar will appear on the screen showing the status and progress of the simulation as shown in the **Figure 1.3** below.

Figure 1.3: Screenshot of the Progress bar.

Results **:** There are 3 options to manipulate the results.

(i) View results: Click on the Plot view radio button in the UI. The Result View GUI is displayed as shown in **the Figure 1.4.**

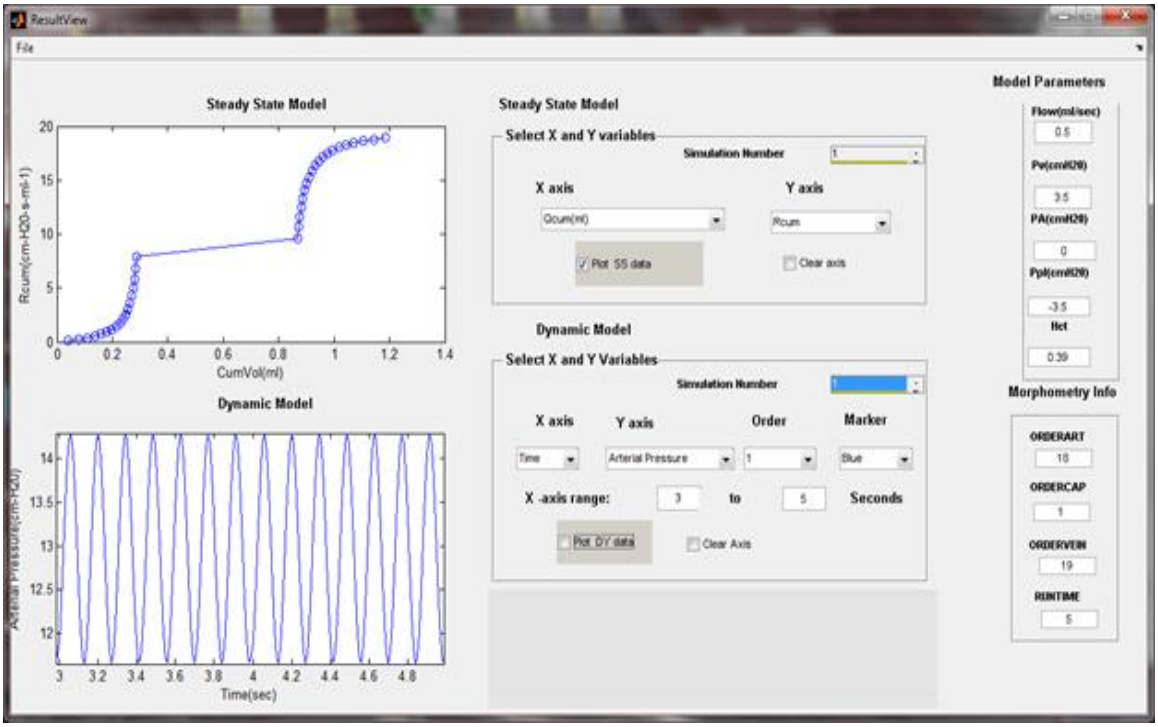


Figure 1.4: Screenshot of the Result View GUI

The right side of the result view GUI has four sections:

- Model Parameters**:** Steady state model parameters including mean flow, venous pressure, alveolar pressure, pleural pressure and hematocrit ratio.
- Morphometry Info**:** Includes the number of orders of arteries, veins and capillaries, and displays run time for the dynamic simulation.
- Steady State Model**:** Option to select steady state simulation result variables for plotting in the result top panel. The x-axis variable options for the steady state simulation plots are i) Cumulative volume (Vcum), and ii) Mean flow (Flow). The y-axis options are i) Mean pressure (Pmid), ii) Cumulative resistance (Rcum), iii) Cumulative Inertance (Lcum), iv) Cumulative Compliance (Ccum), or v) Pulmonary arterial pressure (*Pa*).

  Simulation Number**:** The indexes of the number of simulations run. After selecting the Simulation number and the x- and y- axis variables, the user

checks the plot SS data check box. The results of the selection are plotted in the top panel as shown in the Figure 1.4. In this case, the figure shows cumulative resistance versus cumulative volume. The axis can be cleared using the clear axis check box and can be re-plotted again with different user-selected results.

- Dynamic Model**:** This section is used to select the dynamic simulation result variables, for plotting in the result in the bottom panel. The following are the options for the dynamic model plots.

  Simulation number: The index of the number of simulations run. It consists of a list box to select a result structure from a list of multiple simulation result structures.
  Dynamic X axis variable: The x-axis variable is Time (seconds).

  Dynamic Y axis variables: The y-axis variables can be selected from Pulmonary Arterial pressure, Venous pressure, Capillary Pressure, Arterial Flow, Venous Flow or Cardiac Output.

  Order: Select plots for a particular vessel order. The information on the arterial, venous and capillary orders are shown in the morphometry info section.

  Marker: Plots can be given different symbol colors (red, blue, green, cyan, magenta, yellow and black) using the marker drop down menu icon.

(ii) Export to Excel:
This module is called by clicking the 'Export to excel' radio button in the Result option section of the Simulation UI. The user is prompted to select an excel file to write the simulation results. Then four tabs are created in the excel sheet. **Figure 1.4** shows a screenshot of the excel sheet where the four tabs are:

Simulation Parameters: steady state and dynamic parameters used for the model simulation.

Functional Models: options used for the simulation.

Steady State Simulation Results: steady-state results are organized under the headings Arteries, Capillaries and Veins with the corresponding Simulation name and number.

Dynamic Simulation Results: dynamic results and their values.

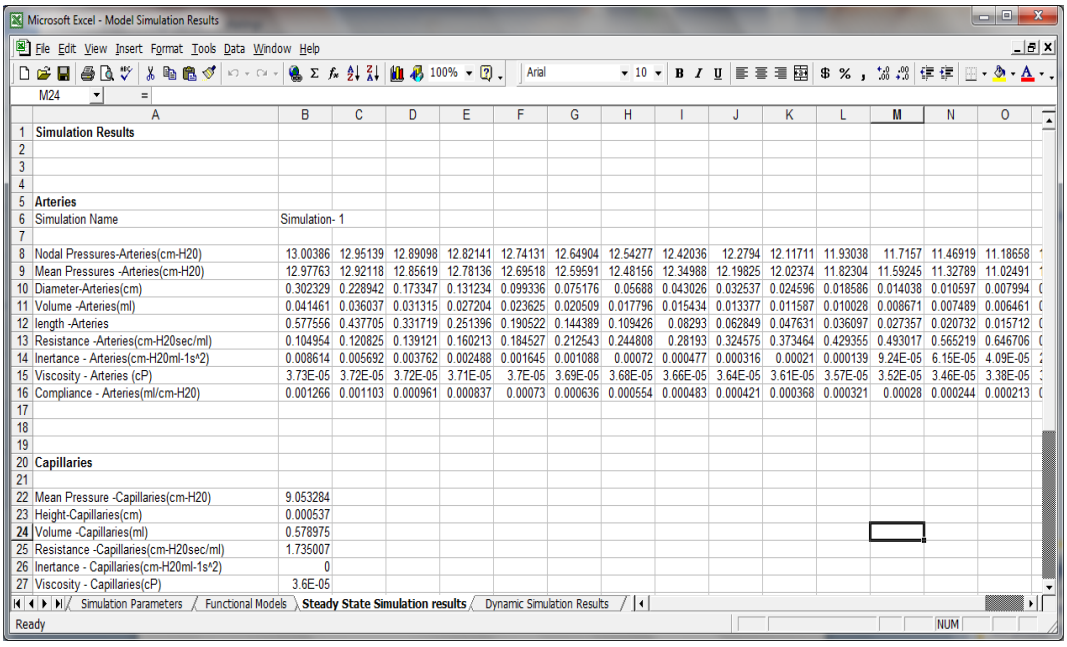


Figure 1.4: Screenshot of the excel sheet with the simulation results. The four tabs Simulation Parameters, Functional Models, Steady State Simulation Results and Dynamic Simulation results are shown. The tab highlighted shows the steady state simulation results.

The function used for this operation is [status] = ExcelExport(gSim_Results). The input to the function is the 'gSim_Results' structure. The output is a status variable; either 1 or -1 based on whether the data was written successfully to the excel sheet or not.

(iii) Save Results

This module is called when the user selects the 'Save Results' radio button in the 'Result Option' section of the Simulation UI. The results and parameters of the model simulation are saved as matlab (.mat) files. The parameter file can be reloaded using the' Load' button in order to rerun the simulation file. The function used for this operation is [status] = Save_Results(gSim_Results). The input to this function is the gSim_Results structure. The output is a status variable; either 1 or -1 based on whether the data was saved successfully to the specified location.

Multiple Simulations:

Multiple simulation options can be done using the Add, and Copy buttons as shown in **Figure 1.5.** A simulation can be deleted using the Delete button. When the user clicks Add, a new structure is created with the default existing parameters and options. The Copy button creates a new structure with the parameters and options of the previous simulation selection. Also, the user can give specific names for each simulation under "Name". A multiple input file with 6 simulations is stored in the multiple input folders in the Parameters folder. The result of Pulmonary arterial pressure vs. flow can be seen by clicking the plot view in the steady state model.

Changing distensibility and number of vessels

This module loads the morphometry file based on the user selection in the Simulation UI and stores it in the 'gSim_Params' structure. For hypothesis testing, the arterial and venous distensibility and the number of arteries can be changed using the Update button in the Simulation UI. This button invokes a new GUI called 'Morphometry_File_Change' as shown in **Figure 4.4**

Distensibility: Once the file is loaded, current values of arterial and venous distensibility are shown and can be edited. The update button will modify the morphometric file used as input to run the simulation.

Arterial Rarefaction: This module is called by clicking the Rarefaction- Arteries checkbox. The user can enter the diameter range and the percentage reduction. For example: entering 50 and 100 in the percentage and diameter edit boxes mean eliminate 50% of the vessels with diameter under 100 microns. The update button will update the morphometric file used as input.

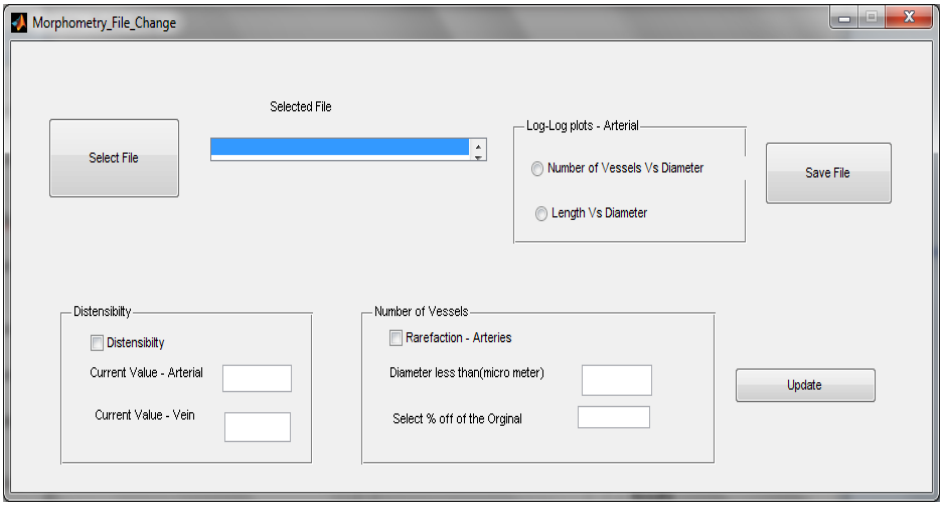


Figure 1.5: GUI for changing the distensibility and number of arteries in the input morphometric file.

Once the options are changed, the model input morphometry parameters are updated and is reflected in the simulation results. The modified MorphData is saved in 'gSim_results{1,1}.Params.MorphData'.

Vasoconstriction Simulation

User can chose from arterial and/or venous vasoconstriction by providing the values in the edit boxes or using the default values.

Occlusion and other dynamic model options

The additional dynamic model options can be found in the 'More Options' GUI, which can be found by clicking the 'More Options' button. The model option section gives the user three dynamic simulation options:
Womersley: uses the womersley equations to calculate resistance.
Variable *RLC*: updates the coefficients during simulation.
Occlusion options: select from i) no occlusion, ii) arterial occlusion, iii) venous occlusion, iv) double occlusion. The duration of the occlusion is entered in the edit box.

d. Matlab Code

This appendix includes the Matlab code of the Lung Model which is organized into Folders Input, Model and Results

1. Folder Input

```
function [] = constants()

% loads initial constants, enumerations

global SIMULATION_TYPE_STEADY SIMULATION_TYPE_DYNAMIC
global MORPHOMETRY_RAT MORPHOMETRY_DOG
global OCCL_NONE OCCL_AO OCCL_DO OCCL_VO OCCL_VODO
global NO_VASO VASO_SMALL VASO_ALL
global MDL_DIAMVSPTM_LIN MDL_DIAMVSPTM_NONLIN
global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
global MDL_CAPHEIGHTVSPTM_NONLIN2
global MDL_PERIVASCULAR_ALBERT
global MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER
global MDL_PERIVASCULAR_BSHOUTY_LAIFOOK
```

```
global MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER
global MDL_PERIVASCULAR_HAWORTH_LAIFOOK
global MDL_LUNGAIRVOL_INFLATION
MDL_LUNGAIRVOL_DEFLATION
global MDL_LENVSVOL_ISOTROPIC MDL_LENVSVOL_SMITHMITZNER
global MDL_CAPAREAVSVOL_TWOTHIRDPOWER
MDL_CAPAREAVSVOL_GILBASED
global MDL_POSTDIAMVSVOL_ISOTROPIC
MDL_POSTDIAMVSVOL_GILBASED
global MDL_POSTDIAMVSVOL_CONSTVOL
global MDL_CAPLENWIDVSVOL_ISOTROPIC
MDL_CAPLENWIDVSVOL_SMITHMITZNER
global MDL_CAPGEOMZONE2_CUSP
MDL_CAPGEOMZONE2_TETHEREDCUSP
global MDL_CAPGEOMZONE2_COLLAPSE
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
global MDL_CAPGEOMZONE2_NONE
global MDL_CARDIACCYCLE_SINUSOIDAL
MDL_CARDIACCYCLE_RECTSINE
global MDL_BREATHINGCYCLE_1
global MDL_VISCOSITY_LINEHAN MDL_VISCOSITY_KIANIHUDETZ
global MDL_HCTFAHRAEUS_ARTVEN
global MDL_GEOMFRICTIONFACTOR_FACTOR
global MDL_MORPHNVSD_NVSD
global MDL_MORPHLVSD_LVSD
global MDL_STRUCTURE_HOMOGENOUS

% simulation parameters
% simulation type
SIMULATION_TYPE_STEADY = 1; SIMULATION_TYPE_DYNAMIC = 2;

% morphometry openfiles/constants
MORPHOMETRY_RAT = 1; MORPHOMETRY_DOG = 2;
MORPHOMETRY_MOUSE = 3;

% occlusions
OCCL_NONE = 1; OCCL_AO = 2; OCCL_DO = 3;
OCCL_VO = 4; OCCL_VODO = 5;

% vasoconstriction
NO_VASO = 1; VASO_SMALL = 2; VASO_ALL = 3;

% functional models
% diameter vs. Ptm
MDL_DIAMVSPTM_LIN = 1; MDL_DIAMVSPTM_NONLIN = 2;
```

```
% capillary height vs. Ptm
MDL_CAPHEIGHTVSPTM_LIN = 1; MDL_CAPHEIGHTVSPTM_NONLIN1
= 2;
MDL_CAPHEIGHTVSPTM_NONLIN2 = 3;

% perivascular pressure
MDL_PERIVASCULAR_ALBERT = 1;
MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER = 2;
MDL_PERIVASCULAR_BSHOUTY_LAIFOOK = 3;
MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER = 4;
MDL_PERIVASCULAR_HAWORTH_LAIFOOK = 5;

% lung air volume
MDL_LUNGAIRVOL_INFLATION = 1; MDL_LUNGAIRVOL_DEFLATION
= 2;

% length vs. volume
MDL_LENVSVOL_ISOTROPIC = 1; MDL_LENVSVOL_SMITHMITZNER =
2;

% capillary area vs. volume
MDL_CAPAREAVSVOL_TWOTHIRDPOWER = 1;
MDL_CAPAREAVSVOL_GILBASED = 2;

% post diameter vs. volume
MDL_POSTDIAMVSVOL_ISOTROPIC = 1;
MDL_POSTDIAMVSVOL_GILBASED = 2;
MDL_POSTDIAMVSVOL_CONSTVOL = 3;

% capillary length/width vs. volume
MDL_CAPLENWIDVSVOL_ISOTROPIC = 1;
MDL_CAPLENWIDVSVOL_SMITHMITZNER = 2;

% capillary sheet geometry zone 2
MDL_CAPGEOMZONE2_CUSP = 1;
MDL_CAPGEOMZONE2_TETHEREDCUSP = 2;
MDL_CAPGEOMZONE2_COLLAPSE = 3;
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS = 4;
MDL_CAPGEOMZONE2_NONE = 5;

% cardiac cycle
MDL_CARDIACCYCLE_SINUSOIDAL = 1;
MDL_CARDIACCYCLE_RECTSINE = 2;
```

```
% breathing cycle
MDL_BREATHINGCYCLE_1 = 1;

% viscosity model
MDL_VISCOSITY_LINEHAN = 1;
MDL_VISCOSITY_KIANIHUDETZ = 2;

% hematocrit / fahraeus effect
MDL_HCTFAHRAEUS_ARTVEN = 1;

% geometric friction factor
MDL_GEOMFRICTIONFACTOR_FACTOR = 1;

% morphometry - N vs. D
MDL_MORPHNVSD_NVSD = 1;

% morphometry - L vs. D
MDL_MORPHLVSD_LVSD = 1;

% structure
MDL_STRUCTURE_HOMOGENOUS = 1;

% morphometric constants

global MORPH_RAT_AREA MORPH_RAT_VSTR
global MORPH_DOG_AREA MORPH_DOG_VSTR
global MORPH_MOUSE_AREA MORPH_MOUSE_VSTR
global DEFAULT_RAT_Q DEFAULT_RAT_HR DEFAULT_RAT_BR
global DEFAULT_DOG_Q DEFAULT_DOG_HR DEFAULT_DOG_BR
global DEFAULT_MOUSE_Q DEFAULT_MOUSE_HR
DEFAULT_MOUSE_BR

MORPH_RAT_AREA = 0.123; % m^2
MORPH_RAT_VSTR = 0.743;

MORPH_DOG_AREA = 27.4; % m^2
MORPH_DOG_VSTR = 0.82;

MORPH_MOUSE_AREA = 0.145; % m^2
MORPH_MOUSE_VSTR = 0.743;

DEFAULT_RAT_Q = 0.5;
DEFAULT_RAT_HR = 7; % hz
DEFAULT_RAT_BR = 1.2; % hz
```

```
DEFAULT_DOG_Q = 42;
DEFAULT_DOG_HR = 2; % hz
DEFAULT_DOG_BR = 0.2; % hz

DEFAULT_MOUSE_Q = 0.5;
DEFAULT_MOUSE_HR = 7; % hz
DEFAULT_MOUSE_BR = 1.2; % hz

% misc. constants

global BLOOD_DENSITY
BLOOD_DENSITY = 1.05;

global FUNG_CAPSHEET_SEGMENTS
FUNG_CAPSHEET_SEGMENTS = 1;%10;

global VESSEL_TYPE_ART VESSEL_TYPE_CAP VESSEL_TYPE_VEN

VESSEL_TYPE_ART = 1;
VESSEL_TYPE_CAP = 2;
VESSEL_TYPE_VEN = 3;

global MIN_VISCOELASTIC_DIAMETER
MIN_VISCOELASTIC_DIAMETER = 0.04; % cm

% functional model constants

% capillary sheet distensibility
global CONSTS_MDL_CAPHEIGHTVSPTM_LIN
global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1
global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2

CONSTS_MDL_CAPHEIGHTVSPTM_LIN.k1 = 0;   % cm-H2O^-1
CONSTS_MDL_CAPHEIGHTVSPTM_LIN.h0inf = 0; % mu-m -> cm
CONSTS_MDL_CAPHEIGHTVSPTM_LIN.h00 = 4.287 / 1e4; % mu-m -> cm
CONSTS_MDL_CAPHEIGHTVSPTM_LIN.alpha = 0.21439348 / 1e4;
CONSTS_MDL_CAPHEIGHTVSPTM_LIN.gamma = 2;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1.k1 = -0.11;   % cm-H2O^-1
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1.h0inf = 1.42 / 1e4;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1.h00 = 3.53 / 1e4;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1.alpha = 0.123;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1.gamma = 3.29;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2.k1 = -0.07;   % cm-H2O^-1
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2.h0inf = 1.42 / 1e4;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2.h00 = 4.22 / 1e4;
```

```
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2.alpha = 0.0565;
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2.gamma = 2.07;

% perivascular pressure
global CONSTS_MDL_PERIVASCULAR

CONSTS_MDL_PERIVASCULAR.BSHOUTY_SMITHMITZNER_ART_ki =...
   [1.577e1 -4.20e-1 -1.495e1 -7.24e-2 -2.03e-3 -1.14e-3];
CONSTS_MDL_PERIVASCULAR.BSHOUTY_SMITHMITZNER_VEN_ki
=... [3.51e1 -7.18e-1 -3.44e1 3.99e-1 -4.12e-4 -8.10e-4];
CONSTS_MDL_PERIVASCULAR.BSHOUTY_LAIFOOK_ART_ki =...
   [6.109e0 -7.363e-1 -7.014e0 1.282e-1 1.02e-2 -1.87e-3];
CONSTS_MDL_PERIVASCULAR.BSHOUTY_LAIFOOK_VEN_ki =...
   [3.63e0 -4.34e-1 -3.01e0 -2.06e-1 -3.69e-3 -4.76e-3];
CONSTS_MDL_PERIVASCULAR.HAWORTH_SMITHMITZNER_ART_ki =
[3.287e0  6.396e0 -1.037e-2 4.322e-1];
CONSTS_MDL_PERIVASCULAR.HAWORTH_SMITHMITZNER_VEN_ki
=[3.24e0 8.42e0 -1.275e-2 3.109e-1];
CONSTS_MDL_PERIVASCULAR.HAWORTH_LAIFOOK_ART_ki = ...
   [3.287 6.396 -.01037 .4322];
CONSTS_MDL_PERIVASCULAR.HAWORTH_LAIFOOK_VEN_ki = ...
   [3.24 8.42 -.01275 .3109];


% lung air volume
global CONSTS_MDL_LUNGAIRVOL

CONSTS_MDL_LUNGAIRVOL.RAT.k1 = 0.11;          % cmH2O^-1
CONSTS_MDL_LUNGAIRVOL.RAT.k2 = 0.1471;        % cmH2O^-1
CONSTS_MDL_LUNGAIRVOL.RAT.Vm = 8.79 + 1;
CONSTS_MDL_LUNGAIRVOL.RAT.Ptp_FRC = 3.5;

CONSTS_MDL_LUNGAIRVOL.DOG.k1 = 0.11;          % cmH2O^-1
CONSTS_MDL_LUNGAIRVOL.DOG.k2 = 0.1471;        % cmH2O^-1
CONSTS_MDL_LUNGAIRVOL.DOG.Vm = 3082 + 331.4;
CONSTS_MDL_LUNGAIRVOL.DOG.Ptp_FRC = 3.5;

% length vs. volume
global CONSTS_MDL_LENVSVOL

CONSTS_MDL_LENVSVOL.SMITHMITZNER_k1 = 0.78;
CONSTS_MDL_LENVSVOL.SMITHMITZNER_k2 = -2.96;

% capillary post diameter vs. volume
global CONSTS_MDL_POSTDIAMVSVOL_GILBASED
```

```
CONSTS_MDL_POSTDIAMVSVOL_GILBASED.k1 = 0.96;
CONSTS_MDL_POSTDIAMVSVOL_GILBASED.k2 = 3.67;

% capillary sheet width/length vs. volume
global CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER

CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER.k1 = 0.97;
CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER.k2 = 3.30;

% capillary sheet zone 2 conditions
global CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS

CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS.hs = 1.417 /
10e4; % mu-m -> cm

% viscosity model
global CONSTS_MDL_VISCOSITY

CONSTS_MDL_VISCOSITY.KIANIHUDETZ.mu_p = 1.7; % cP
CONSTS_MDL_VISCOSITY.KIANIHUDETZ.muc_k1 = 0.48;
CONSTS_MDL_VISCOSITY.KIANIHUDETZ.muc_k2 = 2.35;
CONSTS_MDL_VISCOSITY.KIANIHUDETZ.delta_k1 = 2.03;
CONSTS_MDL_VISCOSITY.KIANIHUDETZ.delta_k2 = 1.0; %
CONSTS_MDL_VISCOSITY.KIANIHUDETZ.Dmin = 2.7;    % mu-m

CONSTS_MDL_VISCOSITY.LINEHAN.mu_p = 1.7; % (cP)
CONSTS_MDL_VISCOSITY.LINEHAN.mua_k1 = 1.97;
CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k1 = 2.90;
CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k2 = 0.28;
CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k3 = 12.04;
CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k4 = 1;

% geometric friction factor

global CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR

CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k1 = -2.28;
CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k2 = 3.18;
CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k3 = -3.58;
CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k4 = 3.65;

%%
function [r] = ModelParams()
% ModelParams - Class to submit Model Parameters for PC Physiome 2.0
```

```
global MORPH_RAT_AREA MORPH_RAT_VSTR
global DEFAULT_RAT_Q DEFAULT_RAT_HR DEFAULT_RAT_BR

% name
r.SimulationName = 'Simulation';

% type
r.SimulationType = 2;

% morphometry
r.Morphometry = [1];
r.UserDefinedMorph = [0];
r.OpenFile = '';
r.MorphData = [];
r.Area = MORPH_RAT_AREA;
r.VSTR = MORPH_RAT_VSTR;

% Rat Steady Parameters
 r.Q = [DEFAULT_RAT_Q];
 r.Pv = 3.5;
 r.PA = 0;
 r.Ppl =-3.5;
 r.Hct =0.45;


% Dynamic Parameters
r.TimeStep = [0.01];
r.RunTime = [5];
r.HeartRate = [DEFAULT_RAT_HR];
r.SysDiasRatio = [0.9];
r.BreathingRate = [DEFAULT_RAT_BR];


% more dynamic parameters
r.WallResistanceX = [10];
r.ViscoCoefRatio = [0.99];
r.ViscoThreshold = [400];
r.WomersleyTheta = [90];
r.LeadInTime = [0];
r.MinPtp = 4;
r.MaxPtp =3.5;

% more options
r.Womersley = [0];
```

```
r.VariableRLC = [0];

% more viscoconstriction changes
r.ArtVaso = [1];
r.ArtBeta1 = [2.35];
r.ArtDRatio = [0.85];
r.ArtAlphaRatio = [1];
r.VenVaso = [1];
r.VenBeta1 = [2.35];
r.VenDRatio = [0.85];
r.VenAlphaRatio = [1];

% occlusion models
r.OcclusionType = [1];
r.AOTime = [];
r.VOTime = [];
r.DOTime = [];

% functional models

r.ArtVen_Distension = [2];
r.Cap_Distension = [2];
r.ArtVen_PerivascularPressure = [5];
r.System_Viscosity = [1];
r.System_LungAirVolume = [2];
r.ArtVen_LengthVsVolume = [2];
r.Cap_LengthWidthVsVolume = [2];
r.Cap_PostDiamVsVolume = [3];
r.Cap_Zone2Geometry = [1];
r.System_CardiacCycle = [1];
r.System_BreathingCycle = [1];
r.System_HctFahreusEffect = [2];
r.Cap_GeometricFrictionFactor = [1];


%%
function [] = GUIToParams(nSimIndex)

global hSim hOpts gSim_Params
global SIMULATION_TYPE_STEADY SIMULATION_TYPE_DYNAMIC
global MORPHOMETRY_RAT MORPHOMETRY_DOG
MORPHOMETRY_MOUSE
global NO_STEPPED STEPPED_Q STEPPED_Pv
global STEPPED_PA STEPPED_Ppl STEPPED_Hct
```

```
global NO_OCCLUSION ART_OCCLUSION VEN_OCCLUSION
DOUBLE_OCCLUSION
global NO_VASO VASO_SMALLART VASO_ALLART VASO_SMALLVEN
VASO_ALLVEN
global MDL_DIAMVSPTM_LIN MDL_DIAMVSPTM_NONLIN
global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
global MDL_CAPHEIGHTVSPTM_NONLIN2
global MDL_PERIVASCULAR_ALBERT
global MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER
global MDL_PERIVASCULAR_BSHOUTY_LAIFOOK
global MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER
global MDL_PERIVASCULAR_HAWORTH_LAIFOOK
global MDL_LUNGAIRVOL_INFLATION
MDL_LUNGAIRVOL_DEFLATION
global MDL_LENVSVOL_ISOTROPIC MDL_LENVSVOL_SMITHMITZNER
global MDL_CAPAREAVSVOL_TWOTHIRDPOWER
MDL_CAPAREAVSVOL_GILBASED
global MDL_POSTDIAMVSVOL_ISOTROPIC
MDL_POSTDIAMVSVOL_GILBASED
global MDL_POSTDIAMVSVOL_CONSTVOL
global MDL_CAPLENWIDVSVOL_ISOTROPIC
MDL_CAPLENWIDVSVOL_SMITHMITZNER
global MDL_CAPGEOMZONE2_CUSP
MDL_CAPGEOMZONE2_TETHEREDCUSP
global MDL_CAPGEOMZONE2_COLLAPSE
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
global MDL_CARDIACCYCLE_SINUSOIDAL
MDL_CARDIACCYCLE_RECTSINE
global MDL_BREATHINGCYCLE_1
global MDL_VISCOSITY_LINEHAN MDL_VISCOSITY_KIANIHUDETZ
global MDL_HCTFAHRAEUS_ARTVEN
global MDL_GEOMFRICTIONFACTOR_FACTOR
global MDL_MORPHNVSD_NVSD
global MDL_MORPHLVSD_LVSD
global MDL_STRUCTURE_HOMOGENOUS

oParams = ModelParams;

% simulation name
oParams.SimulationName = get(hSim.txtSimulationName, 'String');

% simulation type
if (get(hSim.rdoSteady, 'Value') == 1)
   oParams.SimulationType = SIMULATION_TYPE_STEADY;
```

```
elseif (get(hSim.rdoDynamic, 'Value') == 1)
   oParams.SimulationType = SIMULATION_TYPE_DYNAMIC;
end

% morphometry
if (get(hSim.rdoRat, 'Value') == 1)
   oParams.Morphometry = MORPHOMETRY_RAT;
elseif (get(hSim.rdoDog, 'Value') == 1)
   oParams.Morphometry = MORPHOMETRY_DOG;
elseif (get(hSim.rdoMouse, 'Value') == 1)
   oParams.Morphometry = MORPHOMETRY_MOUSE;
end
oParams.UserDefinedMorph = get(hSim.chkUserDefinedMorph, 'Value');
if (oParams.UserDefinedMorph == 1)
   oParams.OpenFile = get(hSim.btnLoadMorph, 'UserData');
else
   switch (oParams.Morphometry)
      case MORPHOMETRY_RAT
         oParams.OpenFile = 'Input\geometry\rat6.openfile';
      case MORPHOMETRY_DOG
         oParams.OpenFile = 'Input\geometry\Dog.openfile';
      case MORPHOMETRY_MOUSE
         oParams.OpenFile = 'Input\geometry\mouse7.openfile';
   end
end

oParams.MorphData = [];
oParams.Area = str2double(get(hSim.txtCapSheetArea, 'String'));
oParams.VSTR = str2double(get(hSim.txtVSTR, 'String'));

% steady
oParams.Q = str2double(get(hSim.txtQ, 'String'));
oParams.Pv = str2double(get(hSim.txtPv, 'String'));
oParams.PA = str2double(get(hSim.txtPA, 'String'));
oParams.Ppl = str2double(get(hSim.txtPpl, 'String'));
oParams.Hct = str2double(get(hSim.txtHct, 'String'));

% dynamic
oParams.TimeStep = str2double(get(hSim.txtTimeStep, 'String'));
oParams.RunTime = str2double(get(hSim.txtRunTime, 'String'));
oParams.HeartRate = str2double(get(hSim.txtHeartRate, 'String'));
oParams.SysDiasRatio = str2double(get(hSim.txtSystoleDiastole, 'String'));
oParams.BreathingRate = str2double(get(hSim.txtBreathingRate, 'String'));
% more dynamic parameters
oParams.WallResistanceX = str2double(get(hOpts.txtWallR, 'String'));
```

```
oParams.ViscoCoefRatio = str2double(get(hOpts.txtViscoCoef, 'String'));
oParams.ViscoThreshold = str2double(get(hOpts.txtViscoThreshold, 'String'));
oParams.WomersleyTheta = str2double(get(hOpts.txtWomersleyTheta, 'String'));
oParams.LeadInTime = str2double(get(hOpts.txtLeadInTime, 'String'));
oParams.MinPtp = str2double(get(hOpts.txtMinPtp, 'String'));
oParams.MaxPtp = str2double(get(hOpts.txtMaxPtp, 'String'));

% more options
if (get(hOpts.chkWomersley, 'Value') == 1)
   oParams.Womersley = 1;
else
   oParams.Womersley = 0;
end
if (get(hOpts.chkVariableRLC, 'Value') == 1)
   oParams.VariableRLC = 1;
else
   oParams.VariableRLC = 0;
end

% more viscoconstriction changes
oParams.ArtVaso = get(hSim.ddlVasoArtery, 'Value');
oParams.ArtBeta1 = str2double(get(hSim.txtVasoArtBeta1, 'String'));
oParams.ArtDRatio = str2double(get(hSim.txtVasoArtDRatio, 'String'));
oParams.ArtAlphaRatio = str2double(get(hSim.txtVasoArtAlphaRatio, 'String'));
oParams.VenVaso = get(hSim.ddlVasoVein, 'Value');
oParams.VenBeta1 = str2double(get(hSim.txtVasoVenBeta1, 'String'));
oParams.VenDRatio = str2double(get(hSim.txtVasoVenDRatio, 'String'));
oParams.VenAlphaRatio = str2double(get(hSim.txtVasoVenAlphaRatio,
'String'));

 % occlusion
global OCCL_NONE OCCL_AO OCCL_DO OCCL_VO OCCL_VODO
oParams.OcclusionType = get(hOpts.ddlOcclusionType, 'Value');
oParams.AOTime = []; oParams.DOTime = []; oParams.VOTime = [];
switch (oParams.OcclusionType)
   case OCCL_NONE
      % nothing
   case OCCL_AO
      oParams.AOTime = str2double(get(hOpts.txtOcclusionTime1, 'String'));
   case OCCL_DO
      oParams.DOTime = str2double(get(hOpts.txtOcclusionTime1, 'String'));
   case OCCL_VO
      oParams.VOTime = str2double(get(hOpts.txtOcclusionTime1, 'String'));
   case OCCL_VODO
      oParams.VOTime = str2double(get(hOpts.txtOcclusionTime1, 'String'));
```

```
    oParams.DOTime = str2double(get(hOpts.txtOcclusionTime2, 'String'));
end

 % functional models

oParams.ArtVen_Distension = get(hOpts.ddlArtVenDistension, 'Value');
oParams.Cap_Distension = get(hOpts.ddlCapDistension, 'Value');
oParams.ArtVen_PerivascularPressure = get(hOpts.ddlArtVenPerivascular,
'Value');
oParams.System_LungAirVolume = get(hOpts.ddlSystemAirVolume, 'Value');
oParams.ArtVen_LengthVsVolume = get(hOpts.ddlArtVenLength, 'Value');
oParams.Cap_LengthWidthVsVolume = get(hOpts.ddlCapLengthWidth, 'Value');
oParams.Cap_PostDiamVsVolume = get(hOpts.ddlCapPostDiam, 'Value');
oParams.Cap_Zone2Geometry = get(hOpts.ddlCapZone2Geometry, 'Value');
oParams.System_CardiacCycle = get(hOpts.ddlCardiacCycle, 'Value');
oParams.System_BreathingCycle = get(hOpts.ddlBreathingCycle, 'Value');
oParams.System_Viscosity = get(hOpts.ddlSystemViscosity, 'Value');
gSim_Params{nSimIndex} = oParams;

%%
function [] = ParamsToGUI(nSimIndex)

global hSim hOpts gSim_Params
global SIMULATION_TYPE_STEADY SIMULATION_TYPE_DYNAMIC
global MORPHOMETRY_RAT MORPHOMETRY_DOG
global NO_STEPPED STEPPED_Q STEPPED_Pv
global STEPPED_PA STEPPED_Ppl STEPPED_Hct
global NO_OCCLUSION ART_OCCLUSION VEN_OCCLUSION
DOUBLE_OCCLUSION
global NO_VASO VASO_SMALLART VASO_ALLART VASO_SMALLVEN
VASO_ALLVEN
global MDL_DIAMVSPTM_LIN MDL_DIAMVSPTM_NONLIN
global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
global MDL_CAPHEIGHTVSPTM_NONLIN2
global MDL_PERIVASCULAR_ALBERT
global MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER
global MDL_PERIVASCULAR_BSHOUTY_LAIFOOK
global MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER
global MDL_PERIVASCULAR_HAWORTH_LAIFOOK
global MDL_LUNGAIRVOL_INFLATION
MDL_LUNGAIRVOL_DEFLATION
global MDL_LENVSVOL_ISOTROPIC MDL_LENVSVOL_SMITHMITZNER
global MDL_CAPAREAVSVOL_TWOTHIRDPOWER
MDL_CAPAREAVSVOL_GILBASED
```

```
global MDL_POSTDIAMVSVOL_ISOTROPIC
MDL_POSTDIAMVSVOL_GILBASED
global MDL_POSTDIAMVSVOL_CONSTVOL
global MDL_CAPLENWIDVSVOL_ISOTROPIC
MDL_CAPLENWIDVSVOL_SMITHMITZNER
global MDL_CAPGEOMZONE2_CUSP
MDL_CAPGEOMZONE2_TETHEREDCUSP
global MDL_CAPGEOMZONE2_COLLAPSE
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
global MDL_CARDIACCYCLE_SINUSOIDAL
MDL_CARDIACCYCLE_RECTSINE
global MDL_BREATHINGCYCLE_1
global MDL_VISCOSITY_LINEHAN MDL_VISCOSITY_KIANIHUDETZ
global MDL_HCTFAHRAEUS_ARTVEN
global MDL_GEOMFRICTIONFACTOR_FACTOR
global MDL_MORPHNVSD_NVSD
global MDL_MORPHLVSD_LVSD
global MDL_STRUCTURE_HOMOGENOUS

oParams = gSim_Params{nSimIndex};

% simulation name
set(hSim.ddlSimulations, 'Value', nSimIndex);
set(hSim.txtSimulationName, 'String', oParams.SimulationName);

% simulation type
switch (oParams.SimulationType)
   case SIMULATION_TYPE_STEADY
      set(hSim.rdoSteady, 'Value', 1);
      set(hSim.txtTimeStep, 'Enable', 'off');
      set(hSim.txtRunTime, 'Enable', 'off');
      set(hSim.txtHeartRate, 'Enable', 'off');
      set(hSim.txtSystoleDiastole, 'Enable', 'off');
      set(hSim.txtBreathingRate, 'Enable', 'off');
      set(hOpts.txtWallR, 'Enable', 'off');
      set(hOpts.txtViscoCoef, 'Enable', 'off');
      set(hOpts.txtWomersleyTheta, 'Enable', 'off');
      set(hOpts.txtLeadInTime, 'Enable', 'off');
   case SIMULATION_TYPE_DYNAMIC
      set(hSim.rdoDynamic, 'Value', 1);
      set(hSim.txtTimeStep, 'Enable', 'on');
      set(hSim.txtRunTime, 'Enable', 'on');
      set(hSim.txtHeartRate, 'Enable', 'on');
      set(hSim.txtSystoleDiastole, 'Enable', 'on');
      set(hSim.txtBreathingRate, 'Enable', 'on');
```

```
        set(hOpts.txtWallR, 'Enable', 'on');
        set(hOpts.txtViscoCoef, 'Enable', 'on');
        set(hOpts.txtWomersleyTheta, 'Enable', 'on');
        set(hOpts.txtLeadInTime, 'Enable', 'on');
        end

% morphometry
switch (oParams.Morphometry)
   case MORPHOMETRY_RAT
      set(hSim.rdoRat, 'Value', 1);
   case MORPHOMETRY_DOG
      set(hSim.rdoDog, 'Value', 1);
   case MORPHOMETRY_MOUSE
      set(hSim.rdoMouse, 'Value', 1);
   otherwise
      set(hSim.rdoRat, 'Value', 1);
end

set(hSim.chkUserDefinedMorph, 'Value', oParams.UserDefinedMorph);
if (oParams.UserDefinedMorph == 1)
   set(hSim.btnLoadMorph, 'UserData', oParams.OpenFile);
end
set(hSim.txtCapSheetArea, 'String', oParams.Area);
set(hSim.txtVSTR, 'String', oParams.VSTR);

% steady
set(hSim.txtQ, 'String', oParams.Q);
set(hSim.txtPv, 'String', oParams.Pv);
set(hSim.txtPA, 'String', oParams.PA);
set(hSim.txtPpl, 'String', oParams.Ppl);
set(hSim.txtHct, 'String', oParams.Hct);

% dynamic
set(hSim.txtTimeStep, 'String', oParams.TimeStep);
set(hSim.txtRunTime, 'String', oParams.RunTime);
set(hSim.txtHeartRate, 'String', oParams.HeartRate);
set(hSim.txtSystoleDiastole, 'String', oParams.SysDiasRatio);
set(hSim.txtBreathingRate, 'String', oParams.BreathingRate);

% more dynamic parameters
set(hOpts.txtWallR, 'String', oParams.WallResistanceX);
set(hOpts.txtViscoCoef, 'String', oParams.ViscoCoefRatio);
set(hOpts.txtViscoThreshold, 'String', oParams.ViscoThreshold);
set(hOpts.txtWomersleyTheta, 'String', oParams.WomersleyTheta);
set(hOpts.txtLeadInTime, 'String', oParams.LeadInTime);
```

```
set(hOpts.txtMinPtp, 'String', oParams.MinPtp);
set(hOpts.txtMaxPtp, 'String', oParams.MaxPtp);

% more options
set(hOpts.chkWomersley, 'Value', oParams.Womersley);
set(hOpts.chkVariableRLC, 'Value', oParams.VariableRLC);

% more viscoconstriction changes
global NO_VASO VASO_SMALL VASO_ALL
set(hSim.ddlVasoArtery, 'Value', oParams.ArtVaso);
set(hSim.txtVasoArtBeta1, 'String', oParams.ArtBeta1);
set(hSim.txtVasoArtDRatio, 'String', oParams.ArtDRatio);
set(hSim.txtVasoArtAlphaRatio, 'String', oParams.ArtAlphaRatio);
switch (oParams.ArtVaso)
   case NO_VASO
      set(hSim.txtVasoArtBeta1, 'Enable', 'off')
      set(hSim.txtVasoArtDRatio, 'Enable', 'off')
      set(hSim.txtVasoArtAlphaRatio, 'Enable', 'off')
   case VASO_SMALL
      set(hSim.txtVasoArtBeta1, 'Enable', 'on')
      set(hSim.txtVasoArtDRatio, 'Enable', 'off')
      set(hSim.txtVasoArtAlphaRatio, 'Enable', 'off')
   case VASO_ALL
      set(hSim.txtVasoArtBeta1, 'Enable', 'off')
      set(hSim.txtVasoArtDRatio, 'Enable', 'on')
      set(hSim.txtVasoArtAlphaRatio, 'Enable', 'on')
end
set(hSim.ddlVasoVein, 'Value', oParams.VenVaso);
set(hSim.txtVasoVenBeta1, 'String', oParams.VenBeta1);
set(hSim.txtVasoVenDRatio, 'String', oParams.VenDRatio);
set(hSim.txtVasoVenAlphaRatio, 'String', oParams.VenAlphaRatio);
switch (oParams.VenVaso)
   case NO_VASO
      set(hSim.txtVasoVenBeta1, 'Enable', 'off')
      set(hSim.txtVasoVenDRatio, 'Enable', 'off')
      set(hSim.txtVasoVenAlphaRatio, 'Enable', 'off')
   case VASO_SMALL
      set(hSim.txtVasoVenBeta1, 'Enable', 'on')
      set(hSim.txtVasoVenDRatio, 'Enable', 'off')
      set(hSim.txtVasoVenAlphaRatio, 'Enable', 'off')
   case VASO_ALL
      set(hSim.txtVasoVenBeta1, 'Enable', 'off')
      set(hSim.txtVasoVenDRatio, 'Enable', 'on')
      set(hSim.txtVasoVenAlphaRatio, 'Enable', 'on')
end
```

```
% occlusion models
global OCCL_NONE OCCL_AO OCCL_DO OCCL_VO OCCL_VODO
set(hOpts.ddlOcclusionType, 'Value', oParams.OcclusionType);
switch (oParams.OcclusionType)
   case OCCL_NONE
      set(hOpts.lblOcclusionTime1, 'Visible', 'off');
      set(hOpts.txtOcclusionTime1, 'Visible', 'off');
      set(hOpts.lblOcclusionTime1Units, 'Visible', 'off');
      set(hOpts.lblOcclusionTime2, 'Visible', 'off');
      set(hOpts.txtOcclusionTime2, 'Visible', 'off');
      set(hOpts.lblOcclusionTime2Units, 'Visible', 'off');
   case OCCL_AO
      set(hOpts.lblOcclusionTime1, 'Visible', 'on');
      set(hOpts.txtOcclusionTime1, 'Visible', 'on');
      set(hOpts.lblOcclusionTime1Units, 'Visible', 'on');
      set(hOpts.lblOcclusionTime2, 'Visible', 'off');
      set(hOpts.txtOcclusionTime2, 'Visible', 'off');
      set(hOpts.lblOcclusionTime2Units, 'Visible', 'off');

      set(hOpts.lblOcclusionTime1, 'String', 'Art. Occlus. Time:');
      set(hOpts.txtOcclusionTime1, 'String', num2str(oParams.AOTime));
   case OCCL_DO
      set(hOpts.lblOcclusionTime1, 'Visible', 'on');
      set(hOpts.txtOcclusionTime1, 'Visible', 'on');
      set(hOpts.lblOcclusionTime1Units, 'Visible', 'on');
      set(hOpts.lblOcclusionTime2, 'Visible', 'off');
      set(hOpts.txtOcclusionTime2, 'Visible', 'off');
      set(hOpts.lblOcclusionTime2Units, 'Visible', 'off');

      set(hOpts.lblOcclusionTime1, 'String', 'Double Occlus. Time:');
      set(hOpts.txtOcclusionTime1, 'String', num2str(oParams.DOTime));
   case OCCL_VO
      set(hOpts.lblOcclusionTime1, 'Visible', 'on');
      set(hOpts.txtOcclusionTime1, 'Visible', 'on');
      set(hOpts.lblOcclusionTime1Units, 'Visible', 'on');
      set(hOpts.lblOcclusionTime2, 'Visible', 'off');
      set(hOpts.txtOcclusionTime2, 'Visible', 'off');
      set(hOpts.lblOcclusionTime2Units, 'Visible', 'off');

      set(hOpts.lblOcclusionTime1, 'String', 'Ven. Occlus. Time:');
      set(hOpts.txtOcclusionTime1, 'String', num2str(oParams.VOTime));
   case OCCL_VODO
      set(hOpts.lblOcclusionTime1, 'Visible', 'on');
      set(hOpts.txtOcclusionTime1, 'Visible', 'on');
```

```
        set(hOpts.lblOcclusionTime1Units, 'Visible', 'on');
        set(hOpts.lblOcclusionTime2, 'Visible', 'on');
        set(hOpts.txtOcclusionTime2, 'Visible', 'on');
        set(hOpts.lblOcclusionTime2Units, 'Visible', 'on');

        set(hOpts.lblOcclusionTime1, 'String', 'Ven. Occlus. Time:');
        set(hOpts.txtOcclusionTime1, 'String', num2str(oParams.VOTime));
        set(hOpts.lblOcclusionTime2, 'String', 'Double Occlus. Time:');
        set(hOpts.txtOcclusionTime2, 'String', num2str(oParams.DOTime));
end

% functional models
set(hOpts.ddlArtVenDistension, 'Value', oParams.ArtVen_Distension);
set(hOpts.ddlCapDistension, 'Value', oParams.Cap_Distension);
set(hOpts.ddlArtVenPerivascular, 'Value',
oParams.ArtVen_PerivascularPressure);
set(hOpts.ddlSystemAirVolume, 'Value', oParams.System_LungAirVolume);
set(hOpts.ddlArtVenLength, 'Value', oParams.ArtVen_LengthVsVolume);
set(hOpts.ddlCapLengthWidth, 'Value', oParams.Cap_LengthWidthVsVolume);
set(hOpts.ddlCapPostDiam, 'Value', oParams.Cap_PostDiamVsVolume);
set(hOpts.ddlCapZone2Geometry, 'Value', oParams.Cap_Zone2Geometry);
set(hOpts.ddlCardiacCycle, 'Value', oParams.System_CardiacCycle);
set(hOpts.ddlBreathingCycle, 'Value', oParams.System_BreathingCycle);
set(hOpts.ddlSystemViscosity, 'Value', oParams.System_Viscosity);
gSim_Params{nSimIndex} = oParams;
```

2. Folder Model

```
function [] = model()
% pc physiome model

% Setting Global Parameters
   global gSim_Params gSim_Results
   global SIMULATION_TYPE_STEADY SIMULATION_TYPE_DYNAMIC
   global total
   global Morph_Result
   global morphchange

   total = 7;

   for nSimIndex = 1:length(gSim_Params)
     % Coping Parameters to oparams

       oParams = gSim_Params{nSimIndex};
```

```
    % Read Morphometric Data
     MorphData = dlmread(oParams.OpenFile);
     MetaMorphData = MorphData(size(MorphData, 1), :);
     if (MetaMorphData(1) ~= -1)
         error('-1', 'improperly formatted openfile')
     end

     gSim_Params{nSimIndex}.MorphData = MorphData;

     % Checking for change in distensibilty
     if Morph_Result.dis_chosen ==100
  gSim_Params{nSimIndex}.MorphData(1:18,5) =
Morph_Result.Updated_alpha_Art(:,1);
    gSim_Params{nSimIndex}.MorphData(20:38,5) =
Morph_Result.Updated_alpha_Vein(:,1);

     end

     % Checking for arterial rarefaction

   if Morph_Result.rar_chosen ==100
  gSim_Params{nSimIndex}.MorphData(1:18,1) =
Morph_Result.Number_Vessels_ART_Modified(:,1);

   end
    % Selection of Simulation Type

      switch (oParams.SimulationType)
         case SIMULATION_TYPE_STEADY
            steadymodel(nSimIndex);
         case SIMULATION_TYPE_DYNAMIC
            steadymodel(nSimIndex);
            dynamicmodel(nSimIndex);
      end
  end
  end

  % Steady Model

   function [] = steadymodel(nSimIndex)

   clear global oParams
   % Global Paramters and Constants
   global oParams gSim_Params gSim_Results total
```

```
    global FUNG_CAPSHEET_SEGMENTS
    global MORPHOMETRY_RAT MORPHOMETRY_DOG
    global VESSEL_TYPE_ART VESSEL_TYPE_CAP VESSEL_TYPE_VEN
    global CONSTS_MDL_LUNGAIRVOL
    global NO_VASO VASO_SMALL VASO_ALL
    global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
    global MDL_CAPHEIGHTVSPTM_NONLIN2
    global CONSTS_MDL_CAPHEIGHTVSPTM_LIN
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2
    global FUNG_CAPSHEET_SEGMENTS
   global Morph_Result
   global morphchange
   oParams = gSim_Params{nSimIndex};

    % storing the parametes and results
    gSim_Results{nSimIndex} = ModelResults;
    gSim_Results{nSimIndex}.Params = oParams;

    % Showing the Progress Bar - 1
     f= 1;
     message1 = 'Steady State Simulation - File Name :';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);

     % Read Paramters

    Q = oParams.Q;
    Pv = oParams.Pv;
    PA = oParams.PA;
    Ppl = oParams.Ppl;
    Hct = oParams.Hct;

    % Morphometry

    MorphData = oParams.MorphData;
    MetaMorphData = MorphData(size(MorphData, 1), :);
    OrdersArt = MetaMorphData(2);
    OrdersCap = MetaMorphData(3);
    OrdersVen = MetaMorphData(4);
    OrdersAll = sum([OrdersArt OrdersCap OrdersVen]);
```

```
gSim_Results{nSimIndex}.ORDERART = OrdersArt;
gSim_Results{nSimIndex}.ORDERCAP = OrdersCap;
gSim_Results{nSimIndex}.ORDERVEN = OrdersVen;

% Showing the Progress Bar - 2
 f = 2;
 message1 = 'Steady State - Loading Morhometry ,File Name :';
 FileNAME = oParams.SimulationName;
 status = strcat(message1,FileNAME);
 progressbarv2(f/total,status);
 display(message1);

% steady model
N_All   = MorphData(1:OrdersAll, 1);
D0_All  = MorphData(1:OrdersAll, 2);
L0_All  = MorphData(1:OrdersAll, 3);
mu_All  = MorphData(1:OrdersAll, 4);
alpha_All = MorphData(1:OrdersAll, 5)./1.36;
gamma_All = MorphData(1:OrdersAll, 6);

% ventilation cycle

gSim_Results{nSimIndex}.AlphaAll = alpha_All;


switch oParams.Morphometry
   case MORPHOMETRY_RAT
      oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.RAT;
   case MORPHOMETRY_DOG
      oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.DOG;
end
k1 = oConstsLungAirVol.k1;
k2 = oConstsLungAirVol.k2;
Vm = oConstsLungAirVol.Vm;
Ptp_FRC = oConstsLungAirVol.Ptp_FRC;


Ptp_i = oParams.MinPtp;
Ptp_e = oParams.MaxPtp;
V_Ptp_i = (1 - k1) * (1 - exp(-k2*Ptp_i)) + k1;
V_Ptp_e = (1 - k1) * (1 - exp(-k2*Ptp_e)) + k1;

CONSTS_MDL_LUNGAIRVOL.Ptp_i = Ptp_i;
CONSTS_MDL_LUNGAIRVOL.Ptp_e = Ptp_e;
CONSTS_MDL_LUNGAIRVOL.V_Ptp_i = V_Ptp_i;
```

```
CONSTS_MDL_LUNGAIRVOL.V_Ptp_e = V_Ptp_e;

clear k1 k2 oConstsLungAirVol CONSTS_MDL_LUNGAIRVOL

 % Showing the Progress Bar - 3
 f = 3;
 message1 = 'Steady State - Venous tree Optimization :';
 FileNAME = oParams.SimulationName;
 status = strcat(message1,FileNAME);
 progressbarv2(f/total,status);
 display(message1);


% venous tree optmization
N   = flipud(N_All(sum([OrdersArt OrdersCap])+1:OrdersAll));
D0  = flipud(D0_All(sum([OrdersArt OrdersCap])+1:OrdersAll));
L0  = flipud(L0_All(sum([OrdersArt OrdersCap])+1:OrdersAll));
mu  = flipud(mu_All(sum([OrdersArt
OrdersCap])+1:OrdersAll))*0.000010197;
alpha = flipud(alpha_All(sum([OrdersArt OrdersCap])+1:OrdersAll));
gamma = flipud(gamma_All(sum([OrdersArt OrdersCap])+1:OrdersAll));
D0_Morph = D0;

if morphchange == 100
   if Morph_Result.dis_chosen ==1
    alpha = Morph_Result.Updated_alpha_Vein(:,1)./1.36;
   end
else
    for i = 1:length(OrdersVen)
     alpha = alpha;
    end
end

gSim_Results{nSimIndex}.AlphaVen = alpha;


% vasoconstriction check

switch (oParams.VenVaso)
   case VASO_SMALL % hypoxia
      b1_c = 2.5;
      b1_h = oParams.VenBeta1;
      b2_c = 1.0;
      b2_h = (b2_c * b1_h) / b1_c;
```

```
                D0 = (D0_Morph .* ((D0_Morph(OrdersVen))^(1-
b1_c/b1_h))).*(D0_Morph.^(1-b1_c/b1_h));
                L0 = L0 .* (1./D0_Morph) .* (D0.^b2_h);

            case VASO_ALL % seratonin
                alpha = alpha * oParams.VenAlphaRatio;
                D0 = D0_Morph * oParams.VenDRatio;
                L0 = L0 .* (1./D0_Morph) .* D0 * (1/oParams.VenDRatio);
        end

        X0 = [Pv ; Pv + cumsum(((128*mu.*L0) ./ (pi*N.*(D0.^4))).*Q) ];
        Pdown = Pv;
        VLB = Pv*ones(length(X0),1); % optimization lower bound - Pv

        options = optimset('TolX', 0.00000001, 'TolCon', 0.00000001, 'TolFun',
0.0000000000001, ...
            'MaxFunEvals', 10000000, 'Algorithm','sqp', ...
            'LargeScale', 'off', 'Display', 'off');


        [Pnod_v, OUTPUT] = fmincon(@FUN_ART_VEIN, X0, [], [], [], [], VLB, ...
            [], [], options, Pdown, Q, Pv, PA, Ppl, Hct, N, D0, L0, ...
            mu, alpha, gamma, VESSEL_TYPE_VEN);

        Ptp = PA - Ppl;

        Pmid_v = (Pnod_v(1:OrdersVen)+Pnod_v(2:OrdersVen+1))/2;

        Px_v = F_Model_PxHat(Pmid_v, Ptp, Ppl, VESSEL_TYPE_VEN);
        Ptm_v = Pmid_v - Px_v;

        [D_v] = F_Model_Diam(D0, Ptm_v, alpha, gamma); % + Px

        [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp);

        [L_v] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

        [mu_a] = F_Model_Viscosity(D_v, Hct);
        mu_a_v = mu_a*0.000010197;

        [V_v] = F_Volume_ArtVen(D_v, L_v, N);
        [R_v] = F_Resistance_ArtVen(D_v, L_v, N, mu_a_v);
        [C_v] = F_Compliance_ArtVen(D_v, D0, L_v, N, alpha, gamma);
        [I_v] = F_Inductance_ArtVen(D_v, L_v, N, mu_a_v);
```

```
    clear D0 L N mu alpha gamma

    % Showing the Progress Bar - 4
     f = 4;
     message1 = 'Steady State - Capillary bed optimization :';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);


%% capillary bed optimization

    Nc  = flipud(N_All(sum([OrdersArt OrdersCap])));
    Lc0 = flipud(L0_All(sum([OrdersArt OrdersCap])));
    mu  = flipud(mu_All(sum([OrdersArt OrdersCap])))*0.000010197;


    switch oParams.Cap_Distension
        case MDL_CAPHEIGHTVSPTM_LIN
            oConstsCapHeightVsPtm = CONSTS_MDL_CAPHEIGHTVSPTM_LIN;
        case MDL_CAPHEIGHTVSPTM_NONLIN1
            oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1;
        case MDL_CAPHEIGHTVSPTM_NONLIN2
            oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2;
    end

     h00 = oConstsCapHeightVsPtm.h00;
    hc = h00;
    alphac = oConstsCapHeightVsPtm.alpha;
    gammac = oConstsCapHeightVsPtm.gamma;
    epsilonc_ref = h00;

    clear oConstsCapHeightVsPtm

    Area_ref = oParams.Area * 100^2;
    VSTR = oParams.VSTR;
    Wc_ref = Area_ref ./ Lc0;
    fc = 2.677;

    Lc_ref = Lc0 * ones(FUNG_CAPSHEET_SEGMENTS,1) /
FUNG_CAPSHEET_SEGMENTS;
    Lc_test= Lc0;
```

```
    X0 = (12*mu.*fc.*(cumsum(Lc_ref)).^2)/(Area_ref * VSTR * hc^3)*Q;

    % Checking Zone 2

   if ((Pv + sum(R_v)*Q) < PA)
       ModeZ2 = 1; % Zone 2 = true
       if (FUNG_CAPSHEET_SEGMENTS > 1)
          X0 = PA + [0 ; X0];
       else
          X0 = PA + X0;
       end
   else % Zone 2 = False
       ModeZ2 = 0;
       if (FUNG_CAPSHEET_SEGMENTS > 1)
          X0 = Pv + sum(R_v)*Q + [0 ; X0];
       else
          X0 = Pv + sum(R_v)*Q + X0;
       end
    end

    Pdown = Pnod_v(OrdersVen+1); % boundary condition
    VLB = Pdown; % optimization lower-bound


    options = optimset('TolX', 0.00000001, 'TolCon', 0.00000001, 'TolFun',
0.0000000000001, ...
       'MaxFunEvals', 10000000, 'Algorithm','sqp', ...
       'LargeScale', 'off', 'Display', 'off');


    [Pnod_c, OUTPUT] = fmincon(@FUN_CAP, X0, [], [], [], [], VLB, ...
       [], [], options, Pdown, Q, Pv, PA, Ppl, Hct, ...
       Area_ref, Lc_ref, Wc_ref, epsilonc_ref, Lc0, alphac, fc, VSTR, ...
       Ptp_FRC, mu, ModeZ2, VESSEL_TYPE_CAP);

    if (FUNG_CAPSHEET_SEGMENTS > 1)
       Pmid_c =
(Pnod_c(1:FUNG_CAPSHEET_SEGMENTS)+Pnod_c(2:FUNG_CAPSHEET_S
EGMENTS+1))/2;
    else
       Pmid_c = Pnod_c;
    end
```

```
    % lung air volume functional model
    Ptp = PA - Ppl;
    [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp);
    V_Ptp = V_Ptp*ones(OrdersCap, 1);
    V_Ptp_ref = V_Ptp_ref*ones(OrdersCap, 1);

    % sheet dimensions functional model
    [Lc, Wc] = F_Model_CapSheet_Length_Width(Lc_ref, Wc_ref, V_Ptp,
V_Ptp_ref);
    Areac = Lc.*Wc;

    % sheet thickness model
    Ptm = Pmid_c + PA;
    [h_c] = F_Model_CapSheet_Height(Ptm, Ptp);
    [hc0FRC] = F_Model_CapSheet_Height(0, Ptp_FRC);

    % viscosity functional model
    [mu_a] = F_Model_Viscosity(h_c, Hct);
    mu_a_c = mu_a*0.000010197;

    % sheet post thickness
    [epsilonc] = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
        V_Ptp, V_Ptp_ref, h_c, hc0FRC);

    % geometric friction factor
    [fc] = F_Model_FrictionFactor(h_c, VSTR, epsilonc);

    [V_c] = F_Volume_Cap(h_c, Areac, VSTR);
    [R_c] = F_Resistance_Cap(h_c, Lc, Areac, fc, mu_a_c, VSTR);
    [C_c] = F_Compliance_Cap(h_c, Areac, h00, alphac, gammac, VSTR);
    [I_c] = F_Inductance_Cap();

    % Showing the Progress Bar - 5
     f = 5;
     message1 = 'Steady State - Arterial tree Optimization :';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);

%% Arterial
    % arterial tree optimization
  N   = flipud(N_All(1:OrdersArt));
  D0  = flipud(D0_All(1:OrdersArt));
  L0  = flipud(L0_All(1:OrdersArt));
```

```
mu  = flipud(mu_All(1:OrdersArt))*0.000010197;
alpha = flipud(alpha_All(1:OrdersArt));
gamma = flipud(gamma_All(1:OrdersArt));
D0_Morph = D0;

 gSim_Results{nSimIndex}.Nart = N;
 if morphchange == 100
  if Morph_Result.dis_chosen ==1
     alpha = Morph_Result.Updated_alpha_Art(:,1)./1.36;
  end
  else
    for i = 1:length(OrdersArt)
    alpha = alpha;
     end
  end


  gSim_Results{nSimIndex}.AlphaArt = alpha;

  % vasoconstriction check

  switch (oParams.ArtVaso)
     case VASO_SMALL % hypoxia
        b1_c = 2.5;
        b1_h = oParams.ArtBeta1;
        b2_c = 1.0;
        b2_h = (b2_c * b1_h) / b1_c;
        D0 = (D0_Morph .* ((D0_Morph(OrdersArt)).^(1-
(b1_c./b1_h))./(D0_Morph.^(1-(b1_c./b1_h))));
        Lref = (L0 .* (1./D0_Morph)) .* (D0.^b2_h);
        L0 = Lref;

        gSim_Results{nSimIndex}.STEADY.VasoArt_small.D0 = flipud(D0)';
        gSim_Results{nSimIndex}.STEADY.VasoArt_small.L0 = flipud(Lref)';
     case VASO_ALL % seratonin
        alpha = alpha * oParams.ArtAlphaRatio;
        D0 = D0_Morph * oParams.ArtDRatio;
        L0 = L0 .* (1./D0_Morph) .* D0 * (1/oParams.ArtDRatio);
        gSim_Results{nSimIndex}.STEADY.VasoArt_All.D0 =flipud(D0);
         gSim_Results{nSimIndex}.STEADY.VasoArt_All.L0 =flipud(L0);
  end

  X0 = [Pv ; Pv + cumsum(((128*mu.*L0) ./ (pi*N.*(D0.^4))).*Q) ];

  if (FUNG_CAPSHEET_SEGMENTS > 1)
```

```
    Pdown = Pnod_c(FUNG_CAPSHEET_SEGMENTS+1);
  else
    Pdown = Pnod_c(1);
  end
  VLB = Pdown*ones(length(X0),1); % optimization lower bound - Pv

  options = optimset('TolX', 0.00000001, 'TolCon', 0.00000001, 'TolFun',
0.0000000000001, ...
    'MaxFunEvals', 10000000, 'Algorithm','sqp', ...
    'LargeScale', 'off', 'Display', 'off');

  [Pnod_a, OUTPUT] = fmincon(@FUN_ART_VEIN, X0, [], [], [], [], VLB, ...
    [], [], options, Pdown, Q, Pv, PA, Ppl, Hct, N, D0, L0, ...
    mu, alpha, gamma, VESSEL_TYPE_ART);

  Ptp = PA - Ppl;

  Pmid_a = (Pnod_a(1:OrdersArt)+Pnod_a(2:OrdersArt+1))/2;

  Px_a = F_Model_PxHat(Pmid_a, Ptp, Ppl, VESSEL_TYPE_ART);
  Ptm_a = Pmid_a - Px_a;

  [D_a] = F_Model_Diam(D0, Ptm_a, alpha, gamma);

  [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp);

  [L_a] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

  [mu_a] = F_Model_Viscosity(D_a, Hct);
  mu_a_a = mu_a*0.000010197; % cmH20sec/cP

  [V_a] = F_Volume_ArtVen(D_a, L_a, N);
  [R_a] = F_Resistance_ArtVen(D_a, L_a, N, mu_a_a);
  [C_a] = F_Compliance_ArtVen(D_a, D0, L_a, N, alpha, gamma);
  [I_a] = F_Inductance_ArtVen(D_a, L_a, N, mu_a_a);

  clear D0 L N mu alpha gamma

   % Showing the Progress Bar - 6
   f = 6;
   message1 = 'Steady State - Saving Results :';
   FileNAME = oParams.SimulationName;
   status = strcat(message1,FileNAME);
   progressbarv2(f/total,status);
   display(message1);
```

```
%% Saving Results

   Steady_PnodePAP = Pnod_a(2);

   gSim_Results{nSimIndex}.STEADY.Pnode =
[flipud(Pnod_a(1:length(Pnod_a))) ; ...
       flipud(Pnod_c) ; flipud(Pnod_v(2:length(Pnod_v)))];
   gSim_Results{nSimIndex}.STEADY.Pmid = [flipud(Pmid_a) ; flipud(Pmid_c)
; flipud(Pmid_v)];
   gSim_Results{nSimIndex}.STEADY.D = [flipud(D_a) ; flipud(h_c) ;
flipud(D_v)];
   gSim_Results{nSimIndex}.STEADY.V = [flipud(V_a) ; flipud(V_c) ;
flipud(V_v)];
   gSim_Results{nSimIndex}.STEADY.L = [flipud(L_a) ; flipud(Lc) ;
flipud(L_v)];
   gSim_Results{nSimIndex}.STEADY.R = [flipud(R_a) ; flipud(R_c) ;
flipud(R_v)];
   gSim_Results{nSimIndex}.STEADY.C = [flipud(C_a) ; flipud(C_c) ;
flipud(C_v)];
   gSim_Results{nSimIndex}.STEADY.I = [flipud(I_a) ; flipud(I_c) ;
flipud(I_v)];
   gSim_Results{nSimIndex}.STEADY.mu = [flipud(mu_a_a) ; flipud(mu_a_c) ;
flipud(mu_a_v)];


Qa = sum(V_a);
Qc = sum(V_c);
Qv = sum(V_v);
cumvol = Qa+Qv+Qc;
Ca =sum(C_a);
Cc = sum(C_c);
Cv = sum(C_v) ;
CumCom = Ca + Cc +Cv;

 La =sum(L_a);
Lc = 0;
Lv = sum(L_v) ;
CumInert = La + Lc +Lv;

Ra =sum(R_a);
Rc = sum(R_c);
Rv = sum(R_v) ;
CumRes = Ra + Rc +Rv;
```

```
    % pressure Pnode
    gSim_Results{nSimIndex}.STEADY.Part = flipud(Pnod_a(2:length(Pnod_a)));
    gSim_Results{nSimIndex}.STEADY.Pcap = flipud(Pnod_c);
    gSim_Results{nSimIndex}.STEADY.Pveins =
flipud(Pnod_v(2:length(Pnod_v)));

    % pressure Pmid
    gSim_Results{nSimIndex}.STEADY.Pmidart = flipud(Pmid_a);
    gSim_Results{nSimIndex}.STEADY.Pmidcap = flipud(Pmid_c);
    gSim_Results{nSimIndex}.STEADY.Pmidvein = flipud(Pmid_v);

    % Volume
    gSim_Results{nSimIndex}.STEADY.VOLart = flipud(V_a);
    gSim_Results{nSimIndex}.STEADY.VOLcap = flipud(V_c);
    gSim_Results{nSimIndex}.STEADY.VOLvein = flipud(V_v);
    gSim_Results{nSimIndex}.STEADY.CUMVOL = cumvol;

    % Diameter
    gSim_Results{nSimIndex}.STEADY.Dart = flipud(D_a);
    gSim_Results{nSimIndex}.STEADY.Hcap = flipud(h_c);
    gSim_Results{nSimIndex}.STEADY.Dveins = flipud(D_v);

    % Inductance

    gSim_Results{nSimIndex}.STEADY.Lart = flipud(L_a);
    gSim_Results{nSimIndex}.STEADY.Lcap = flipud(Lc);
    gSim_Results{nSimIndex}.STEADY.Lveins =flipud(L_v);

     % resistance

    gSim_Results{nSimIndex}.STEADY.Rart =flipud(R_a);
    gSim_Results{nSimIndex}.STEADY.Rcap = flipud(R_c);
    gSim_Results{nSimIndex}.STEADY.Rveins = flipud(R_v);
    gSim_Results{nSimIndex}.STEADY.CUMRES = CumRes;
     % Compliance

    gSim_Results{nSimIndex}.STEADY.Cart =flipud(C_a);
    gSim_Results{nSimIndex}.STEADY.Ccap = flipud(C_c);
    gSim_Results{nSimIndex}.STEADY.Cveins = flipud(C_v);
    gSim_Results{nSimIndex}.STEADY.CUMCOM = CumCom;

     % Inertance

    gSim_Results{nSimIndex}.STEADY.Iart = flipud(I_a);
```

```
    gSim_Results{nSimIndex}.STEADY.Icap = flipud(I_c);
    gSim_Results{nSimIndex}.STEADY.Iveins = flipud(I_v);
    % MU

    gSim_Results{nSimIndex}.STEADY.MUart = flipud(mu_a_a);
    gSim_Results{nSimIndex}.STEADY.MUcap = flipud(mu_a_c);
    gSim_Results{nSimIndex}.STEADY.MUveins =flipud(mu_a_v);

    % PAP
    for i = 1:length(gSim_Results)

   Q(i) = gSim_Results{1,i}.Params.Q ;
    PAP(i) = gSim_Results{1,i}.STEADY.Pmidart(1,1) ;
    PV(i) = gSim_Results{1,i}.Params.Pv ;
   end
gSim_Results{1,1}.MeanFlow = Q;
gSim_Results{1,1}.PAP = PAP;
gSim_Results{1,1}.PV = PV;



     % Showing the Progress Bar - 7
     f = 7;
     message1 = 'Steady State Simulation Complete :';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);
  gSim_Results{nSimIndex}.Params = gSim_Params{nSimIndex};

%

function [F, GG] = FUN_ART_VEIN(PRES, Pdown, Q, Pv, PA, Ppl, Hctf, ...
    N, D0, L0, mu, alpha, gamma, VesselType)
% function FUN_ART_VEIN solves the Poiseuille Pressure equations
% for the vessel tree

    Orders = length(PRES) - 1;
    PRES(1) = Pdown;

    Pin  = PRES(1:Orders);
    Pout = PRES(2:Orders+1);
    P = (Pin + Pout) / 2;
    Ptp = PA - Ppl;
```

```
    % lung air volume functional model
    [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp);

    % perivascular pressure functional model
    Px = F_Model_PxHat(P, Ptp, Ppl, VesselType);
    Ptm = P - Px;

    % diameter vs. Ptm functional model
    [D] = F_Model_Diam(D0, Ptm, alpha, gamma);

    % apparent viscosity functional model
    [mu_a] = F_Model_Viscosity(D, Hctf);
    mu_a = mu_a*0.000010197;

    % length vs. volume functional model
    [L] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

    Ploss = ((128*mu_a.*L) ./ (pi*N.*D.^4)) .* Q;

    F = sum(abs(Pout - Pin - Ploss));
    GG = -1;


function [F,GG] = FUN_CAP(PRES, Pdown, Q, Pv, PA, Ppl, Hctf, ...
    Area_ref, Lc_ref, Wc_ref, epsilonc_ref, Lc0, alpha, fc, VSTR, ...
    Ptp_FRC, mu, ModeZ2, VesselType)
% optimization script for determining capillary bed pressures

    global oParams
    global MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
    global CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
    global CONSTS_MDL_VISCOSITY
    global FUNG_CAPSHEET_SEGMENTS

    % check fung segmentation
    OrdersCap = length(PRES) - 1;

    if (ModeZ2 ~= 0)
       PfinalZ3 = PA;
    else
       PfinalZ3 = Pdown;
    end

    if (OrdersCap >= 1)
       % PRES(1) = PfinalZ3;
```

```
        Pin = PRES(1:OrdersCap);
        Pout = PRES(2:OrdersCap+1);
    else
        Pin = PfinalZ3;
        Pout = PRES(1);
    end

    % lung air volume functional model
    Ptp = PA - Ppl;
    [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp);
    V_Ptp = V_Ptp;%*ones(OrdersCap, 1);
    V_Ptp_ref = V_Ptp_ref;%*ones(OrdersCap, 1);

    % sheet dimensions functional model
    [Lc, Wc] = F_Model_CapSheet_Length_Width(Lc_ref, Wc_ref, V_Ptp,
V_Ptp_ref);

    % sheet thickness model
    Ptm = (Pout + Pin)/2 + PA;
    [hc] = F_Model_CapSheet_Height(Ptm, Ptp);
    [hc0FRC] = F_Model_CapSheet_Height(0, Ptp_FRC);
    [hc0Ptp] = F_Model_CapSheet_Height(0, Ptp);

    % viscosity functional model
    [mu_a] = F_Model_Viscosity(hc, Hctf);
    mu_a = mu_a*0.000010197;

    % sheet post thickness
    [epsilonc] = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
        V_Ptp, V_Ptp_ref, hc, hc0FRC);

    % geometric friction factor
    [fc] = F_Model_FrictionFactor(hc, VSTR, epsilonc);

    Area = sum(Lc .* Wc);

    if (ModeZ2 ~= 0 && ...
            oParams.Cap_Zone2Geometry ==
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS)
        hs_Z2 =
CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS.hs;
        mu_Z2 = CONSTS_MDL_VISCOSITY.LINEHAN.mua_k1*0.000010197;
        epsilonc_Z2 = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
            V_Ptp, V_Ptp_ref, hs_Z2, hc0FRC);
        fc_Z2 = F_Model_FrictionFactor(hs_Z2, VSTR, epsilonc_Z2);
```

```
        DeltaP_Z2 = PA - Pdown;
        Lc_Z2 = (DeltaP_Z2*Wc.*hs_Z2.^3) ./ (12*Q*mu_Z2.*fc_Z2);
        Lc_Z3 = (sum(Lc) - Lc_Z2) * ...

ones(FUNG_CAPSHEET_SEGMENTS,1)/FUNG_CAPSHEET_SEGMENTS;
        Area_Z3 = Lc_Z3 * Wc;


    else
        Lc_Z3 = Lc;
        Area_Z3 = Area;
    end

    Ploss =
(12*mu_a.*fc.*(Lc_Z3.^2))./((Area_Z3/FUNG_CAPSHEET_SEGMENTS) *
VSTR .* hc.^3)*Q;

    F = sum(abs(Pout - Pin - Ploss));
    GG = -1;

 %Dynamic Model

%% Dynamic Simulation

 function [] = dynamicmodel(nSimIndex)
     total = 7;

    global oParams gSim_Params gSim_Results
    global FUNG_CAPSHEET_SEGMENTS
    global VESSEL_TYPE_ART VESSEL_TYPE_CAP VESSEL_TYPE_VEN
    global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
    global MDL_CAPHEIGHTVSPTM_NONLIN2
    global CONSTS_MDL_CAPHEIGHTVSPTM_LIN
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2
    global MIN_VISCOELASTIC_DIAMETER
    global MORPHOMETRY_RAT MORPHOMETRY_DOG
    global CONSTS_MDL_LUNGAIRVOL



    oParams = gSim_Params{nSimIndex};

    Q = gSim_Params{nSimIndex}.Q;
```

```
Pv = gSim_Params{nSimIndex}.Pv;
PA = gSim_Params{nSimIndex}.PA;
Ppl = gSim_Params{nSimIndex}.Ppl;
Hct = gSim_Params{nSimIndex}.Hct;

% Showing the Progress Bar - 1
 f = 1;
 message1 = 'Dynamic Simulation - Reading Parameters : ';
 FileNAME = oParams.SimulationName;
 status = strcat(message1,FileNAME);
 progressbarv2(f/total,status);
 display(message1);



MorphData = oParams.MorphData;
MetaMorphData = MorphData(size(MorphData, 1), :);
OrdersArt = MetaMorphData(2);
% OrdersCap = MetaMorphData(3);
OrdersCap = FUNG_CAPSHEET_SEGMENTS;
OrdersVen = MetaMorphData(4);
OrdersAll = sum([OrdersArt 1 OrdersVen]);
OrdersAllWCaps = sum([OrdersArt OrdersCap OrdersVen]);

% Showing the Progress Bar - 2
 f = 2;
 message1 = 'Dynamic Simulation : Reading Morhometry Data :';
 FileNAME = oParams.SimulationName;
 status = strcat(message1,FileNAME);
 progressbarv2(f/total,status);
 display(message1);



N_All   = MorphData(1:OrdersAll, 1);
D0_All  = MorphData(1:OrdersAll, 2);
L0_All  = MorphData(1:OrdersAll, 3);
mu_All  = MorphData(1:OrdersAll, 4);
alpha_All = MorphData(1:OrdersAll, 5);
gamma_All = MorphData(1:OrdersAll, 6);

tspan = [[-oParams.LeadInTime:oParams.TimeStep:-oParams.TimeStep]
[0:oParams.TimeStep:oParams.RunTime]];

DynArtNodes = OrdersArt * 3;
```

```
   DynCapNodes = OrdersCap * 2;
   DynVenNodes = (OrdersVen+1) * 3;

   % viscoelasticity check


   switch oParams.Cap_Distension
      case MDL_CAPHEIGHTVSPTM_LIN
         oConstsCapHeightVsPtm = CONSTS_MDL_CAPHEIGHTVSPTM_LIN;
      case MDL_CAPHEIGHTVSPTM_NONLIN1
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1;
      case MDL_CAPHEIGHTVSPTM_NONLIN2
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2;
   end
   hc = oConstsCapHeightVsPtm.h00;
   clear oConstsCapHeightVsPtm

   N_All = MorphData(1:OrdersAll, 1);
   D0_All  = MorphData(1:OrdersAll, 2);
   D0_a = D0_All(1:OrdersArt);
   h_c = hc*ones(OrdersCap, 1);
   D0_v = D0_All(OrdersArt+1+1:OrdersAll);
   D0s = [D0_a ; h_c ; D0_v];
   L0_All  = MorphData(1:OrdersAll, 3);
   mu_All  = MorphData(1:OrdersAll, 4);
   alpha_All = MorphData(1:OrdersAll, 5);
   gamma_All = MorphData(1:OrdersAll, 6);



   InputVisco = oParams.ViscoCoefRatio;
   ViscoCoefs = ((D0s >
MIN_VISCOELASTIC_DIAMETER*ones(length(D0s),1))*InputVisco/2) + ...
      InputVisco/2;


% Showing the Progress Bar - 3
    f = 3;
    message1 = 'Dynamic simulation - Setting up arrays to store results :';
    FileNAME = oParams.SimulationName;
    status = strcat(message1,FileNAME);
    progressbarv2(f/total,status);
    display(message1);
```

```
    % setup index arrays
    Art_Index = 1:OrdersArt;
    Cap_Index = OrdersArt+1:OrdersArt+OrdersCap;
    Ven_Index = sum([OrdersArt OrdersCap])+1: ...
       sum([OrdersArt OrdersCap OrdersVen]);
    All_Index = 1:sum([OrdersArt OrdersCap OrdersVen]);

    IndexArtStart = 1;
    IndexArtEnd = 3*OrdersArt;
    IndexCapStart = 3*OrdersArt + 1;
    IndexCapEnd = sum([3*OrdersArt 2*(OrdersCap+1)]);
    IndexVenStart = sum([3*OrdersArt 2*(OrdersCap+1)]) + 1;
    IndexVenEnd = sum([3*OrdersArt 2*(OrdersCap+1) 3*OrdersVen]);

    Pa_Index = IndexArtStart:3:IndexArtEnd-2;
    Pwa_Index = IndexArtStart+1:3:IndexArtEnd-1;
    Qa_Index = IndexArtStart+2:3:IndexArtEnd;

    Pc_Index = IndexCapStart:2:IndexCapEnd-1;
    Pwc_Index = IndexCapStart+1:2:IndexCapEnd;
    %Qc_Index = IndexCapStart+2:IndexArtEnd; %loolu added

    Qv_Index = IndexVenStart:3:IndexVenEnd-2;
    Pv_Index = IndexVenStart+1:3:IndexVenEnd-1;
    Pwv_Index = IndexVenStart+2:3:IndexVenEnd;

    P_Initial = flipud([Pv ;
cumsum(flipud(gSim_Results{nSimIndex}.STEADY.R(All_Index))) ...
        * Q + Pv]);


    % Showing the Progress Bar - 4
     f = 4;
     message1 = 'Dynamic Simulation - Loading Arterial,Capillary and Venous
Steady State Pressures and Flows : ';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);


    % arterial initial conditions
```

```
    Y0(Pa_Index) = P_Initial(Art_Index);
    Y0(Pwa_Index) = P_Initial(Art_Index);
    Y0(Qa_Index) = Q;

    % capillary initial conditions
    Y0(Pc_Index(1:OrdersCap)) = P_Initial(Cap_Index);
    Y0(Pwc_Index(1:OrdersCap)) = P_Initial(Cap_Index);

    Y0(Pc_Index(OrdersCap+1)) = P_Initial(max(Cap_Index)+1);
    Y0(Pwc_Index(OrdersCap+1)) = P_Initial(max(Cap_Index)+1);
   % Y0(Qc_Index) = Q; %looluadded

    % venous initial conditions
    Y0(Qv_Index) = Q;
    Y0(Pv_Index) = P_Initial(Ven_Index+1);
    Y0(Pwv_Index) = P_Initial(Ven_Index+1);


    Ri = gSim_Results{nSimIndex}.STEADY.R;
    Ci = gSim_Results{nSimIndex}.STEADY.C;
    Ii = gSim_Results{nSimIndex}.STEADY.I;
    Vi = gSim_Results{nSimIndex}.STEADY.V;


    options = odeset('RelTol', 1e-4, 'AbsTol', 1e-8, 'NormControl', 'on');

%Showing the Progress Bar - 5
     f = 5;
     message1 = 'Dynamic Simulation - Calculating the Dynamic Vascular Pressure
and Flow using ode15s : ';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);


    [t, Y] = ode15s(@DYNAMIC_ODES_3, tspan, Y0, options, ...
        OrdersArt, OrdersCap, OrdersVen, ...
        Q, Pv, PA, Ppl, Hct, ...
        N_All, D0_All, L0_All, mu_All, alpha_All, gamma_All, ...
        Ri, Ci, Ii, Vi, ViscoCoefs);

    t = t'; Y = Y';

    gSim_Results{nSimIndex}.DYNAMIC.t = t;
```

```
gSim_Results{nSimIndex}.DYNAMIC.Y = Y;



IndexArtStart = 1;
IndexArtEnd = 3*OrdersArt;
IndexCapStart = 3*OrdersArt + 1;
IndexCapEnd = sum([3*OrdersArt 2*(OrdersCap+1)]);
IndexVenStart = sum([3*OrdersArt 2*(OrdersCap+1)]) + 1;
IndexVenEnd = sum([3*OrdersArt 2*(OrdersCap+1) 3*OrdersVen]);

Pa_Index = IndexArtStart:3:IndexArtEnd-2;
Pwa_Index = IndexArtStart+1:3:IndexArtEnd-1;
Qa_Index = IndexArtStart+2:3:IndexArtEnd;

Pc_Index = IndexCapStart:2:IndexCapEnd-1;
Pwc_Index = IndexCapStart+1:2:IndexCapEnd;
Qc_Index = IndexCapStart+2:1:IndexCapEnd;
Qv_Index = IndexVenStart:3:IndexVenEnd-2;
Pv_Index = IndexVenStart+1:3:IndexVenEnd-1;
Pwv_Index = IndexVenStart+2:3:IndexVenEnd;

% breathing cycle


switch gSim_Params{nSimIndex}.Morphometry
   case MORPHOMETRY_RAT
      oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.RAT;
   case MORPHOMETRY_DOG
      oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.DOG;
end
k1 = oConstsLungAirVol.k1;
k2 = oConstsLungAirVol.k2;
Vm = oConstsLungAirVol.Vm;
Ptp_FRC = oConstsLungAirVol.Ptp_FRC;

clear oConstsLungAirVol

Ptp_i = CONSTS_MDL_LUNGAIRVOL.Ptp_i;
Ptp_e = CONSTS_MDL_LUNGAIRVOL.Ptp_e;
V_Ptp_i = CONSTS_MDL_LUNGAIRVOL.V_Ptp_i;
V_Ptp_e = CONSTS_MDL_LUNGAIRVOL.V_Ptp_e;

fb = oParams.BreathingRate;
```

```
phi = -pi/2;
Ptp = Ptp_e + (1/2)*(Ptp_i - Ptp_e) * (1 + sin(2*pi*fb*t + phi));
dPtp = sign(cos(2*pi*fb*t + phi));

V_Ptp = zeros(1, size(Ptp, 2));
% inflation/deflation portion of curve check
V_Ptp(find(dPtp>=0)) = Vm*(((V_Ptp_e-V_Ptp_i)/(Ptp_e-
Ptp_i))*(Ptp(find(dPtp>=0))-Ptp_e) + V_Ptp_e);
V_Ptp(find(dPtp<0)) = Vm * ((1 - k1) * (1 - exp(-k2*Ptp(find(dPtp<0)))) +
k1);
V_Ptp_ref = Vm * ((1 - k1) * (1 - exp(-k2*Ptp_FRC)) + k1);

% arterial

N   = N_All(1:OrdersArt);
D0  = D0_All(1:OrdersArt);
L0  = L0_All(1:OrdersArt);
mu  = mu_All(1:OrdersArt)*0.000010197;
alpha = alpha_All(1:OrdersArt);
gamma = gamma_All(1:OrdersArt);

[Pnode_a] = Y(Pa_Index(1:OrdersArt), :);
[Pmid_a] = (Y(Pa_Index(1:OrdersArt), :) + ...
   Y([Pa_Index(2:OrdersArt) Pc_Index(1)], :)) / 2;

Px_a = F_Model_PxHat(Pmid_a, Ptp, Ppl, VESSEL_TYPE_ART);
Ptm_a = Pmid_a - Px_a;

[D_a] = F_Model_Diam(D0, Ptm_a, alpha, gamma);

[L_a] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

[mu_a] = F_Model_Viscosity(D_a, Hct);
mu_a_a = mu_a*0.000010197;

[V_a] = F_Volume_ArtVen(D_a, L_a, N);
[R_a] = F_Resistance_ArtVen(D_a, L_a, N, mu_a_a);
[C_a] = F_Compliance_ArtVen(D_a, D0, L_a, N, alpha, gamma);
[I_a] = F_Inductance_ArtVen(D_a, L_a, N, mu_a_a);

% capillary


Nc  = N_All(sum([OrdersArt 1]));
Lc0 = L0_All(sum([OrdersArt 1]));
```

```
   mu  = mu_All(sum([OrdersArt 1]))*0.000010197;

   switch oParams.Cap_Distension
      case MDL_CAPHEIGHTVSPTM_LIN
         oConstsCapHeightVsPtm = CONSTS_MDL_CAPHEIGHTVSPTM_LIN;
      case MDL_CAPHEIGHTVSPTM_NONLIN1
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1;
      case MDL_CAPHEIGHTVSPTM_NONLIN2
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2;
   end

   h00 = oConstsCapHeightVsPtm.h00;
   hc = h00;
   alphac = oConstsCapHeightVsPtm.alpha;
   gammac = oConstsCapHeightVsPtm.gamma;
   epsilonc_ref = h00;

   clear oConstsCapHeightVsPtm

   Area_ref = oParams.Area * 100^2;
   VSTR = oParams.VSTR;
   Wc_ref = (sum(Area_ref) ./ Lc0) * ones(FUNG_CAPSHEET_SEGMENTS,
1);
   fc = 2.677;

   Lc_ref = Lc0 * ones(FUNG_CAPSHEET_SEGMENTS,1) /
FUNG_CAPSHEET_SEGMENTS;

   % Lc_ref = Lc0;
   [Pnode_c] = Y(Pc_Index(1:OrdersCap+1), :);
   [Pmid_c] = (Y(Pc_Index(1:OrdersCap), :) + ...
      Y([Pc_Index(2:OrdersCap+1)], :)) / 2;

   % sheet dimensions functional model
   [Lc, Wc] = F_Model_CapSheet_Length_Width(Lc_ref, Wc_ref, V_Ptp,
V_Ptp_ref);
   Areac = Lc.*Wc;

   % sheet thickness model
   Ptm = Pmid_c + PA;
   [h_c] = F_Model_CapSheet_Height(Ptm, Ptp);
   [hc0FRC] = F_Model_CapSheet_Height(0, Ptp_FRC);
```

```
% viscosity functional model
[mu_a] = F_Model_Viscosity(h_c, Hct);
mu_a_c = mu_a*0.000010197;

% sheet post thickness
[epsilonc] = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
   V_Ptp, V_Ptp_ref, h_c, hc0FRC);

% geometric friction factor
[fc] = F_Model_FrictionFactor(h_c, VSTR, epsilonc);

[V_c] = F_Volume_Cap(h_c, Areac, VSTR);
[R_c] = F_Resistance_Cap(h_c, Lc, Areac, fc, mu_a_c, VSTR);
[C_c] = F_Compliance_Cap(h_c, Areac, h00, alphac, gammac, VSTR);
[I_c] = F_Inductance_Cap();

% venous
N   = N_All(sum([OrdersArt 1])+1:OrdersAll);
D0  = D0_All(sum([OrdersArt 1])+1:OrdersAll);
L0  = L0_All(sum([OrdersArt 1])+1:OrdersAll);
mu  = mu_All(sum([OrdersArt 1])+1:OrdersAll)*0.000010197;
alpha = alpha_All(sum([OrdersArt 1])+1:OrdersAll);
gamma = gamma_All(sum([OrdersArt 1])+1:OrdersAll);


[Pnode_v] = Y(Pv_Index(1:OrdersVen), :);
[Pmid_v] = (Y(Pv_Index(1:OrdersVen), :) + ...
   [Y(Pv_Index(2:OrdersVen), :) ; Pv*ones(1,size(Y,2))]) / 2;

Px_v = F_Model_PxHat(Pmid_v, Ptp, Ppl, VESSEL_TYPE_VEN);
Ptm_v = Pmid_v - Px_v;

[D_v] = F_Model_Diam(D0, Pmid_v, alpha, gamma); % + Px

[L_v] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

[mu_a] = F_Model_Viscosity(D_v, Hct);
mu_a_v = mu_a*0.000010197;

[V_v] = F_Volume_ArtVen(D_v, L_v, N);
[R_v] = F_Resistance_ArtVen(D_v, L_v, N, mu_a_v);
[C_v] = F_Compliance_ArtVen(D_v, D0, L_v, N, alpha, gamma);
[I_v] = F_Inductance_ArtVen(D_v, L_v, N, mu_a_v);

[Qa] = (Y(Qa_Index(1:OrdersArt), :));
```

```
    [Qv] = (Y(Qv_Index(1:OrdersVen), :));
    [Qc] = (Y(Qc_Index(1:OrdersCap), :));

    % calculate Flow across capillaries

     % Showing the Progress Bar - 6
     f = 6;
     message1 = 'Dynamic Simulation - Storing Simulation Results :';
     FileNAME = oParams.SimulationName;
     status = strcat(message1,FileNAME);
     progressbarv2(f/total,status);
     display(message1);


    gSim_Results{nSimIndex}.DYNAMIC.Ptp = Ptp;
    gSim_Results{nSimIndex}.DYNAMIC.V_Ptp = V_Ptp;
    gSim_Results{nSimIndex}.DYNAMIC.Pnode = [Pnode_a ; Pnode_c ;
Pnode_v];
    gSim_Results{nSimIndex}.DYNAMIC.Pmid = [Pmid_a ; Pmid_c ; Pmid_v];
    gSim_Results{nSimIndex}.DYNAMIC.Ptm = [Ptm_a ; Ptm ; Ptm_v];
    gSim_Results{nSimIndex}.DYNAMIC.Q = [Qa ;
zeros(FUNG_CAPSHEET_SEGMENTS,size(Qa,2)); Qv];
    gSim_Results{nSimIndex}.DYNAMIC.D =  [D_a ; h_c ; D_v ];
    gSim_Results{nSimIndex}.DYNAMIC.L = [L_a ; Lc ; L_v ];
    gSim_Results{nSimIndex}.DYNAMIC.mu = [mu_a_a ; mu_a_c ; mu_a_v ];
    gSim_Results{nSimIndex}.DYNAMIC.V = [V_a ; V_c ; V_v];
    gSim_Results{nSimIndex}.DYNAMIC.R = [R_a ; R_c ; R_v];
    gSim_Results{nSimIndex}.DYNAMIC.C = [C_a ; C_c ; C_v];
    gSim_Results{nSimIndex}.DYNAMIC.I = [I_a ; I_c*ones(1,size(Y,2)) ; I_v];

    % organizing Results

gSim_Results{nSimIndex}.DYNAMIC.Pnodeart= Pnode_a;
gSim_Results{nSimIndex}.DYNAMIC.Pnodecap= Pnode_c;
gSim_Results{nSimIndex}.DYNAMIC.Pnodevein= Pnode_v;

gSim_Results{nSimIndex}.DYNAMIC.Pmidart= Pmid_a;
gSim_Results{nSimIndex}.DYNAMIC.Pmidcap= Pmid_c;
gSim_Results{nSimIndex}.DYNAMIC.Pmidvein= Pmid_v;

gSim_Results{nSimIndex}.DYNAMIC.Ptmart= Ptm_a;
gSim_Results{nSimIndex}.DYNAMIC.Ptmcap= Ptm;
gSim_Results{nSimIndex}.DYNAMIC.Ptmvein= Ptm_v;

gSim_Results{nSimIndex}.DYNAMIC.Flowart= Qa;
```

```
gSim_Results{nSimIndex}.DYNAMIC.Flowcap=
zeros(FUNG_CAPSHEET_SEGMENTS,size(Qa,2));
gSim_Results{nSimIndex}.DYNAMIC.Flowvein= Qv;

gSim_Results{nSimIndex}.DYNAMIC.Dart= D_a;
gSim_Results{nSimIndex}.DYNAMIC.Hcap= h_c;
gSim_Results{nSimIndex}.DYNAMIC.Dvein= D_v;

gSim_Results{nSimIndex}.DYNAMIC.Lart= L_a;
gSim_Results{nSimIndex}.DYNAMIC.Lcap= Lc;
gSim_Results{nSimIndex}.DYNAMIC.Lvein= L_v;

gSim_Results{nSimIndex}.DYNAMIC.MUart= mu_a_a;
gSim_Results{nSimIndex}.DYNAMIC.MUcap= mu_a_c;
gSim_Results{nSimIndex}.DYNAMIC.MUvein= mu_a_v;

gSim_Results{nSimIndex}.DYNAMIC.VOLart= V_a;
gSim_Results{nSimIndex}.DYNAMIC.VOLcap= V_c;
gSim_Results{nSimIndex}.DYNAMIC.VOLvein= V_v;


gSim_Results{nSimIndex}.DYNAMIC.Rart= R_a;
gSim_Results{nSimIndex}.DYNAMIC.Rcap= R_c;
gSim_Results{nSimIndex}.DYNAMIC.Rvein= R_v;

gSim_Results{nSimIndex}.DYNAMIC.Cart= C_a;
gSim_Results{nSimIndex}.DYNAMIC.Ccap= C_c;
gSim_Results{nSimIndex}.DYNAMIC.Cvein= C_v;

gSim_Results{nSimIndex}.DYNAMIC.Iart= I_a;
gSim_Results{nSimIndex}.DYNAMIC.Icap= I_c*ones(1,size(Y,2));
gSim_Results{nSimIndex}.DYNAMIC.Ivein= I_v;

% Saving Cardiac Output
for i = 1:length(gSim_Results)
Params = gSim_Params{1,i};
RT = gSim_Params{1,i}.RunTime;
Step = gSim_Params{1,i}.TimeStep;
Time = 0:Step:RT;
Qmean = gSim_Params{1,i}.Q;

%%
HR = Params.HeartRate;
t = Time;
Qmean = Qmean;
```

```
if Params.System_CardiacCycle == 1


          Q = Qmean*(1+sin(2*pi*HR*t));
        gSim_Results{1,i}.DYNAMIC.CO = Q;
else
           % Loolu added - March 08 2011
      s_d = Params.SysDiasRatio;
      shift = 1./HR;

 ts = (floor(t ./ (1./HR))) .* (1 ./ HR);
 tnew  = t - ts;
 coef = (tnew <= ((shift .* s_d) ./ 2));
Q = (Qmean./ s_d) .* pi .* ( sin((2 .* pi .* HR .* tnew) ./ s_d)) .* coef;
  gSim_Results{1,i}.DYNAMIC.CO = Q;
end


%%
end

    f = 7;
    message1 = 'Dynamic Simulation :Complete and results are saved';
    FileNAME = oParams.SimulationName;
    status = strcat(message1,FileNAME);
    progressbarv2(f/total,status);
    display(message1);


%% Display if the Dynamic Simulation reached quasi SS ( Comparing the mean
pressures and flow of last
%% 2 and previous 2 seconds)
RunTime = oParams.RunTime;
Timestep = oParams.TimeStep;
display('Simulation Run Time is : ');
display(RunTime);

% last 2 seconds
% Mean of the flow and pressures  Last2 seconds

RM2 = RunTime-2;
No = RM2/Timestep;
Nototal = RunTime/Timestep;

% Flow mean for last 2 seconds
```

```
Qa1 = Qa(1,:);
Qv1 = Qv(1,:);
MF_A = mean(Qa1(No: Nototal));
MF_V = mean(Qv1(No: Nototal));

% Pressures for the last 2 seconds
Pa1 = Pnode_a(1,:);
Pvfinal = Pnode_v(1,:);
MP_A = mean(Pa1(No: Nototal));
MP_V = mean(Pvfinal(No: Nototal));

% Previous 2 seconds before the last 2 seconds

RM3 = RM2-2;
No = RM3/Timestep;
Nototal = RM2/Timestep;

% Flow mean for last 2 seconds
Qa1 = Qa(1,:);
Qv1 = Qv(1,:);
MF_A2 = mean(Qa1(No: Nototal));
MF_V2 = mean(Qv1(No: Nototal));

% Pressures for the last 2 seconds
Pa1 = Pnode_a(1,:);
Pvfinal = Pnode_v(1,:);
MP_A2 = mean(Pa1(No: Nototal));
MP_V2 = mean(Pvfinal(No: Nototal));

gSim_Results{nSimIndex}.DYNAMIC.StatusMF_A1 = MF_A;
gSim_Results{nSimIndex}.DYNAMIC.StatusMF_V1 = MF_V;
gSim_Results{nSimIndex}.DYNAMIC.StatusMF_A2 = MF_A2;
gSim_Results{nSimIndex}.DYNAMIC.StatusMF_V2 = MF_V2;
gSim_Results{nSimIndex}.DYNAMIC.StatusMP_A1 = MP_A;
gSim_Results{nSimIndex}.DYNAMIC.StatusMP_V1 = MP_V;
gSim_Results{nSimIndex}.DYNAMIC.StatusMP_A2 = MP_A2;
gSim_Results{nSimIndex}.DYNAMIC.StatusMP_V2 = MP_V2;

Filename = gSim_Params{nSimIndex}.SimulationName;

title ='Status';
msg1 = ': Simulation Complete';
file = Filename;
message = strcat(file,msg1);
msgbox(message,title);
```

```
 function [dY] = DYNAMIC_ODES_3(t, Y, ...
    OrdersArt, OrdersCap, OrdersVen, ...
   Qmean, Pv, PA, Ppl, Hct, N, D0, L0, mu, alpha, gamma, Ri, Ci, Ii, Vi,
ViscoCoefs)
% the ODE file for the dynamic model
   global oParams
   global FUNG_CAPSHEET_SEGMENTS

   persistent Ra Rc Rv Ca Cc Cv Ia Ic Iv
   persistent Volume
   global YPrev VolPrev


%% Occlusion - AO,VO,DO
ON = 0;
 OA = 0;
 OD= 0;
 OV = 0;
 OVD = 0;

  if (oParams.OcclusionType == 1)
  ON = 1;
  end
  if (oParams.OcclusionType == 2)
   OA = 1;
    AOT = oParams.AOTime;
  end
  if (oParams.OcclusionType == 3)
   OD = 1;
   DOT = oParams.DOTime;
  end
  if (oParams.OcclusionType == 4)
   OV = 1;
   VOT = oParams.VOTime;
  end
 if (oParams.OcclusionType ==5)
   OVD = 5;
   VOT = oParams.VOTime;
   DOT =oParams.DOTime;
 end

   %
   OrdersCap = FUNG_CAPSHEET_SEGMENTS;
```

```
OrdersArtSegments = 1:OrdersArt;
OrdersCapSegments = OrdersArt+1:sum([OrdersArt OrdersCap]);
OrdersVenSegments = sum([OrdersArt OrdersCap])+1: ...
   sum([OrdersArt OrdersCap OrdersVen]);

DynArtNodes = OrdersArt * 3;
DynCapNodes = OrdersCap * 2;
DynVenNodes = (OrdersVen+1) * 3;

IndexArtStart = 1;
IndexArtEnd = 3*OrdersArt;
IndexCapStart = 3*OrdersArt + 1;
IndexCapEnd = sum([3*OrdersArt 2*(OrdersCap+1)]);
IndexVenStart = sum([3*OrdersArt 2*(OrdersCap+1)]) + 1;
IndexVenEnd = sum([3*OrdersArt 2*(OrdersCap+1) 3*OrdersVen]);

if (t+oParams.LeadInTime == 0)
   Ra = Ri(OrdersArtSegments);
   Rc = Ri(OrdersCapSegments);
   Rv = Ri(OrdersVenSegments);

   Ca = Ci(OrdersArtSegments);
   Cc = Ci(OrdersCapSegments);
   Cv = Ci(OrdersVenSegments);

   Ia = Ii(OrdersArtSegments);
   Ic = Ii(OrdersCapSegments);
   Iv = Ii(OrdersVenSegments);

   Q_CO = Qmean;

   Volume = Vi;

   YPrev = Y;
   VolPrev = Volume;
else

   Q_CO = Cardiac_Output(t, Qmean);

   [Ptp, V_Ptp, Volume, RaNew, RcNew, RvNew, CaNew, CcNew, CvNew,
IaNew, IcNew, IvNew] = ...
      Update(t, Y, OrdersArt, OrdersCap, OrdersVen, ...
      N, D0, L0, mu, alpha, gamma, Volume, [Ca ; Cc ; Cv], ViscoCoefs, Hct);
```

```
        if (oParams.VariableRLC ~= 0) & (t+oParams.LeadInTime >= 10e-10)
            Ra = RaNew;
            Rc = RcNew;
            Rv = RvNew;
            Ca = CaNew;
            Cc = CcNew;
            Cv = CvNew;
            Ia = IaNew;
            Ic = IcNew;
            Iv = IvNew;
        end

    end

    WallResistance = oParams.WallResistanceX;

    Rw = (sum([Ra;Rc;Rv]) * WallResistance);

if (OA == 1) &(OD ==0)
        if (t >= AOT)
            AO_Time = 1;
            DO_Time = 1;
            Q_CO = 0;
        Y(3) = 0;
        Y(6) = 0;
        else
         AO_Time = 1;
          DO_Time = 1;
        end
elseif (OD == 1)&(OA == 0)
    if (t >= DOT)

            DO_Time = 0;
            Q_CO = 0;
        Y(3) = 0;
        Y(6) = 0;
        else
         DO_Time = 1;

    end
 elseif (OV == 1)&(OA == 0)
    if (t >= VOT)&(t < DOT)

            DO_Time = 0;
            Q_CO = Qmean;
```

```
    else(t >= DOT)
      Q_CO = 0;

    end
elseif (OV ==1)&(OA ==1)
   if (t >= VOT)&(t < AOT)

        DO_Time = 0;
        Q_CO = Qmean;
    else(t >= AOT)
      Q_CO = 0;

   end
end

   dY = zeros(length(Y),1);

   Pa_Index = IndexArtStart:3:IndexArtEnd-2;
   Pwa_Index = IndexArtStart+1:3:IndexArtEnd-1;
   Qa_Index = IndexArtStart+2:3:IndexArtEnd;

   Pc_Index = IndexCapStart:2:IndexCapEnd-1;
   Pwc_Index = IndexCapStart+1:2:IndexCapEnd;

   Qv_Index = IndexVenStart:3:IndexVenEnd-2;
   Pv_Index = IndexVenStart+1:3:IndexVenEnd-1;
   Pwv_Index = IndexVenStart+2:3:IndexVenEnd;


   Cda = Ca.*ViscoCoefs(OrdersArtSegments);
   Cwa = Ca.*(1-ViscoCoefs(OrdersArtSegments));

   Cdc = Cc.*ViscoCoefs(OrdersCapSegments);
   Cwc = Cc.*(1-ViscoCoefs(OrdersCapSegments));

   Cdv = Cv.*ViscoCoefs(OrdersVenSegments);
   Cwv = Cv.*(1-ViscoCoefs(OrdersVenSegments));
   Rv_out = (Pv - 0.97*Pv)/Qmean;


   % arterial
   % dP{n}/dt = (1/C{n})*(Q{n-1} - Q{n} - (P{n} - Pw{n})/Rw{n})
   % dPw{n}/dt = (1/Cw{n})*((P{n} - Pw{n})/Rw{n})
   % dQ{n}/dt = (1/L{n})*(P{n} - P{n+1} - R{n}*Q{n})
```

```
    dY(Pa_Index) = (1./Cda).*([Q_CO ; Y(Qa_Index(1:OrdersArt-1))] ...
       - Y(Qa_Index) - (Y(Pa_Index) - Y(Pwa_Index))/Rw);

    dY(Pwa_Index) = (1./Cwa).*((Y(Pa_Index) - Y(Pwa_Index))/Rw);

    dY(Qa_Index) = (1./Ia).*(Y(Pa_Index) ...
       - Y([Pa_Index(2:OrdersArt) Pc_Index(1)]) ...
       - Ra(1:OrdersArt) .* Y(Qa_Index));

    % capillary
    % first segment
    % dP{n}/dt = (2/C{n})*(Q{n-1} - (P{n} - P{n+1})/R{n})
    % last segment
    % dP{n}/dt = (2/C{n})*((P{n-1} - P{n})/R{n-1} - Q{n+1})
    % other segments
    % dP{n}/dt = (2/C{n})*((P{n-1} - P{n})/R{n-1} ...
    %     - (P{n} - P{n+1})/R{n})
    % dPw{n}/dt = (2/Cw{n})*((P{n} - Pw{n})/Rw{n})

    if (numel(Pc_Index) == 2)
    % dY(Pc_Index) = (2./Cdc) .* ...
    %     (Y(Qa_Index(OrdersArt)) - Y(Qv_Index(1)));

    dY(Pc_Index(1)) = (2./Cdc) .* (Y(Qa_Index(OrdersArt)) - ...
       (Y(Pc_Index(1)) - Y(Pc_Index(2))) ./ Rc(1));
    else
    dY(Pc_Index(1:OrdersCap)) = (2./Cdc).*([Y(Qa_Index(OrdersArt)) ; ...
       (Y(Pc_Index(1:OrdersCap-1)) - Y(Pc_Index(2:OrdersCap))) ...
       ./ Rc(1:OrdersCap-1)] ...
       - (Y(Pc_Index(1:OrdersCap)) - Y(Pc_Index(2:OrdersCap+1))) ...
       ./ Rc(1:OrdersCap));
    end

    dY(Pwc_Index(1:OrdersCap)) = (2./Cwc).*((Y(Pc_Index(1:OrdersCap)) -
Y(Pwc_Index(1:OrdersCap))) ./ Rw);

    % last index of capillary - new!
    dY(Pc_Index(OrdersCap+1)) = (2./Cdc(OrdersCap)) .* ...
       ((Y(Pc_Index(OrdersCap)) - Y(Pc_Index(OrdersCap+1))) ...
       ./ Rc(OrdersCap) - Y(Qv_Index(1)));

    dY(Pwc_Index(OrdersCap+1)) =
(2./Cwc(OrdersCap)).*((Y(Pc_Index(OrdersCap+1)) -
Y(Pwc_Index(OrdersCap+1))) ./ Rw);
```

```
    % venous
    % dQ{n}/dt = (1/L{n})*(P{n-1} - P{n} - R{n}*Q{n})
    % dP{n}/dt = (1/C{n})*(Q{n} - Q{n+1} - (P{n} - Pw{n})/Rw{n})
    % dPw{n}/dt = (1/Cw{n})*((P{n} - Pw{n})/Rw{n})

    dY(Qv_Index) = (1./Iv).*(Y([Pc_Index(OrdersCap+1) ...
        Pv_Index(1:OrdersVen-1)]) ...
        - Y(Pv_Index) - Rv(1:OrdersVen) .* Y(Qv_Index));

    dY(Pv_Index) = (1./Cdv).*(Y(Qv_Index(1:OrdersVen)) ...
        - [Y(Qv_Index(2:OrdersVen)) ...
        - (Y(Pv_Index(1:OrdersVen-1)) - Y(Pwv_Index(1:OrdersVen-1))) ...
        / Rw ; (Y(Pv_Index(OrdersVen)) - 0.97*Pv)./Rv_out]);

%     dY(Pv_Index) = (1./Cdv).*(Y(Qv_Index(1:OrdersVen)) ...
%         - [Y(Qv_Index(2:OrdersVen)) ...
%         - (Y(Pv_Index(1:OrdersVen-1)) - Y(Pwv_Index(1:OrdersVen-1))) / Rw ...
%         ; (Y(Pv_Index(OrdersVen)) - Y(Pwv_Index(OrdersVen)))./Rv_out]);

    dY(Pwv_Index) = (1./Cwv).*((Y(Pv_Index) - Y(Pwv_Index)) / Rw);


function [Ptp, V_Ptp, Volume, RaOut, RcOut, RvOut, CaOut, CcOut, CvOut, ...
        IaOut, IcOut, IvOut] = Update(t, Y, OrdersArt, OrdersCap, OrdersVen, ...
        N_All, D0_All, L0_All, mu_All, alpha_All, gamma_All, Volume, C,
ViscoCoefs, Hctf)

    %global gSim_Debug
    global oParams
    global MORPHOMETRY_RAT MORPHOMETRY_DOG
    global CONSTS_MDL_LUNGAIRVOL
    global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
    global MDL_CAPHEIGHTVSPTM_NONLIN2
    global CONSTS_MDL_CAPHEIGHTVSPTM_LIN
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2
    global FUNG_CAPSHEET_SEGMENTS
    global MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
    global YPrev VolPrev

    OrdersArtSegments = 1:OrdersArt;
    OrdersCapSegments = OrdersArt+1:sum([OrdersArt OrdersCap]);
    OrdersVenSegments = sum([OrdersArt OrdersCap])+1+1: ...
        sum([OrdersArt OrdersCap OrdersVen])+1;
```

```
    IndexP = [[1:3:3*OrdersArt-2] ...
       [3*OrdersArt+1:2:3*OrdersArt + 2*(OrdersCap+1)-1] ...
       [3*OrdersArt + 2*(OrdersCap+1):3: ...
       3*OrdersArt + 2*(OrdersCap+1) + 3*OrdersVen-2]];
    IndexPw = [[2:3:3*OrdersArt-1] ...
       [3*OrdersArt+2:2:3*OrdersArt + 2*(OrdersCap+1)] ...
       [3*OrdersArt+2*(OrdersCap+1)+2:3: ...
       3*OrdersArt+2*(OrdersCap+1) + 3*OrdersVen-1]];

    if (t+oParams.LeadInTime == 0)
       YPrev = Y;
       VolPrev = Volume;
    end

%     P_now = Y(IndexP);
%     P_prior = YPrev(IndexP);
%     Pw_now = Y(IndexPw);
%     Pw_prior = YPrev(IndexPw);
    LenP = numel(IndexP); LenPw = numel(IndexPw);

    P_now = (Y(IndexP(1:LenP-1)) + Y(IndexP(2:LenP)))/2;
    P_prior = (YPrev(IndexP(1:LenP-1)) + YPrev(IndexP(2:LenP)))/2;
    Pw_now = (Y(IndexPw(1:LenPw-1)) + Y(IndexPw(2:LenPw)))/2;
    Pw_prior = (YPrev(IndexPw(1:LenPw-1)) + YPrev(IndexPw(2:LenPw)))/2;

    % choose species
    switch oParams.Morphometry
       case MORPHOMETRY_RAT
          oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.RAT;
       case MORPHOMETRY_DOG
          oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.DOG;
    end
    k1 = oConstsLungAirVol.k1;
    k2 = oConstsLungAirVol.k2;
    Vm = oConstsLungAirVol.Vm;
    Ptp_FRC = oConstsLungAirVol.Ptp_FRC;

    clear oConstsLungAirVol

    Ptp_i = CONSTS_MDL_LUNGAIRVOL.Ptp_i;
    Ptp_e = CONSTS_MDL_LUNGAIRVOL.Ptp_e;
    V_Ptp_i = CONSTS_MDL_LUNGAIRVOL.V_Ptp_i;
    V_Ptp_e = CONSTS_MDL_LUNGAIRVOL.V_Ptp_e;
```

```
            % transpulmonary pressure
            fb = oParams.BreathingRate;
            phi = -pi/2;
            Ptp = Ptp_e + (1/2)*(Ptp_i - Ptp_e) * (1 + sin(2*pi*fb*t + phi));
            dPtp = sign(cos(2*pi*fb*t + phi));

            global gk
            gk = gk + 1;
            % inflation/deflation portion of curve check
            if (dPtp >= 0)
               % inflation
               % V_Ptp = Vm*(((V_Ptp_e-V_Ptp_i)/(Ptp_e-Ptp_i))*(Ptp-Ptp_e) +
V_Ptp_e);
               V_Ptp = Vm * ((1 - k1) * (1 - exp(-k2*Ptp)) + k1);
            elseif (dPtp < 0)
               % deflation
               V_Ptp = Vm * ((1 - k1) * (1 - exp(-k2*Ptp)) + k1);
            end
            V_Ptp_ref = Vm * ((1 - k1) * (1 - exp(-k2*Ptp_FRC)) + k1);

            Volume_All = VolPrev + C .* (ViscoCoefs .* (P_now - P_prior) ...
               + (1-ViscoCoefs) .* (Pw_now - Pw_prior));
            Volume = Volume_All;

            % recalculate coefficients
            % arterial
            N   = N_All(1:OrdersArt);
            D0  = D0_All(1:OrdersArt);
            L0  = L0_All(1:OrdersArt);
            mu  = mu_All(1:OrdersArt)*0.000010197;
            alpha = alpha_All(1:OrdersArt);
            gamma = gamma_All(1:OrdersArt);

            Volume_a = Volume_All(1:OrdersArt);

            % calculate new vessel lengths
            [L_a] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

            % solve for diameter
            [D_a] = sqrt(4*Volume_a./(pi*L_a.*N));

            % find apparent viscosity
            [mu_a] = F_Model_Viscosity(D_a, Hctf);
            mu_a_a = mu_a*0.000010197;
```

```
   [R_a] = F_Resistance_ArtVen(D_a, L_a, N, mu_a_a);
   [C_a] = F_Compliance_ArtVen(D_a, D0, L_a, N, alpha, gamma);
   [I_a] = F_Inductance_ArtVen(D_a, L_a, N, mu_a_a);

   % capillary

   Nc  = flipud(N_All(sum([OrdersArt 1])));
   Lc0 = flipud(L0_All(sum([OrdersArt 1])));
   mu  = flipud(mu_All(sum([OrdersArt 1])))*0.000010197;

   switch oParams.Cap_Distension
      case MDL_CAPHEIGHTVSPTM_LIN
         oConstsCapHeightVsPtm = CONSTS_MDL_CAPHEIGHTVSPTM_LIN;
      case MDL_CAPHEIGHTVSPTM_NONLIN1
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1;
      case MDL_CAPHEIGHTVSPTM_NONLIN2
         oConstsCapHeightVsPtm =
CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2;
   end

   h00 = oConstsCapHeightVsPtm.h00;
   alphac = oConstsCapHeightVsPtm.alpha;
   gammac = oConstsCapHeightVsPtm.gamma;
   epsilonc_ref = h00;

   clear oConstsCapHeightVsPtm

   Area_ref = oParams.Area * 100^2 ...
      / FUNG_CAPSHEET_SEGMENTS *
ones(FUNG_CAPSHEET_SEGMENTS, 1) ;
   VSTR = oParams.VSTR;
   Wc_ref = (sum(Area_ref) ./ Lc0) * ones(FUNG_CAPSHEET_SEGMENTS,
1);
   fc = 2.677;
   Lc_ref = Lc0 * ones(FUNG_CAPSHEET_SEGMENTS,1) /
FUNG_CAPSHEET_SEGMENTS;

   % length / width of capillary bed
   [Lc, Wc] = F_Model_CapSheet_Length_Width(Lc_ref, Wc_ref, V_Ptp,
V_Ptp_ref);

   % thickness model
   [hc0FRC] = F_Model_CapSheet_Height(0, Ptp_FRC);
   [hc0Ptp] = F_Model_CapSheet_Height(0, Ptp);
```

```
   Area = Lc .* Wc;
   [hc] = Volume_All(OrdersCapSegments) ./ Area;

   % functional model - viscosity
   [mu_a] = F_Model_Viscosity(hc, Hctf);
   mu_a_c = mu_a*0.000010197;

   % sheet post thickness
   [epsilonc] = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
      V_Ptp, V_Ptp_ref, hc, hc0FRC);

   % geometric friction factor
   [fc] = F_Model_FrictionFactor(hc, VSTR, epsilonc);

   % zone 2 investigation
   if (Y(IndexP(sum([OrdersArt FUNG_CAPSHEET_SEGMENTS])+1)) < Ptp)
&& ...
         (oParams.Cap_Zone2Geometry ==
MDL_CAPGEOMZONE2_FINITEMINTHICKNESS)

   global CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS
   global CONSTS_MDL_VISCOSITY


      hs_Z2 =
CONSTS_MDL_CAPGEOMZONE2_FINITEMINTHICKNESS.hs;
      mu_Z2 = CONSTS_MDL_VISCOSITY.LINEHAN.mua_k1*0.000010197;
      epsilonc_Z2 = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
         V_Ptp, V_Ptp_ref, hs_Z2, hc0FRC);
      fc_Z2 = F_Model_FrictionFactor(hs_Z2, VSTR, epsilonc_Z2);
      DeltaP_Z2 = Ptp -
Y(IndexP(OrdersArt+FUNG_CAPSHEET_SEGMENTS+1));
      Lc_Z2 = (DeltaP_Z2*Wc(FUNG_CAPSHEET_SEGMENTS).*hs_Z2.^3) ./
(12*Y(3*OrdersArt)*mu_Z2.*fc_Z2);
      Lc_Z3 = (sum(Lc) - Lc_Z2) * ...

ones(FUNG_CAPSHEET_SEGMENTS,1)/FUNG_CAPSHEET_SEGMENTS;
      Area_Z3 = Lc_Z3 .* Wc;
   else
      Lc_Z3 = Lc;
      Area_Z3 = Area;
   end

   [R_c] = F_Resistance_Cap(hc, Lc_Z3, Area_Z3, fc, mu_a_c, VSTR);
```

```
[C_c] = F_Compliance_Cap(hc, Area_Z3, h00, alphac, gammac, VSTR);
[I_c] = F_Inductance_Cap();

% venous
N   = N_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]));
D0  = D0_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]));
L0  = L0_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]));
mu  = mu_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]))*0.000010197;
alpha = alpha_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]));
gamma = gamma_All(sum([OrdersArt 1])+1: ...
   sum([OrdersArt 1 OrdersVen]));

Volume_v = Volume_All(sum([OrdersArt OrdersCap])+1: ...
   sum([OrdersArt OrdersCap OrdersVen]));

% calculate new vessel lengths
[L_v] = F_Model_Length(L0, V_Ptp, V_Ptp_ref);

% solve for diameter
[D_v] = sqrt(4*Volume_v./(pi*L_v.*N));

% find apparent viscosity
[mu_a] = F_Model_Viscosity(D_v, Hctf);
mu_a_v = mu_a*0.000010197;

[R_v] = F_Resistance_ArtVen(D_v, L_v, N, mu_a_v);
[C_v] = F_Compliance_ArtVen(D_v, D0, L_v, N, alpha, gamma);
[I_v] = F_Inductance_ArtVen(D_v, L_v, N, mu_a_v);

% consolidate coefficients
RaOut = R_a; RcOut = R_c; RvOut = R_v;
CaOut = C_a; CcOut = C_c; CvOut = C_v;
IaOut = I_a; IcOut = I_c; IvOut = I_v;

% save previous values
YPrev = Y;
VolPrev = Volume;
```

```
%Functional Models

function [C] = F_Compliance_ArtVen(D, D0, L, N, alpha, gamma)
% calculates compliance of a vessel tree

   Dsize = size(D,1); tsize = size(D,2);
   C = zeros(Dsize, tsize);
   for k = 1:tsize
      C(:,k) = (pi*L(:,k).*N.*alpha.*D(:,k).*(gamma.*D0 - D(:,k))) ./ (2*(gamma-
1));
   End

function [C] = F_Compliance_Cap(hc, Areac, h00, alphac, gammac, VSTR)
% calculates compliance of the capillary sheet as function of capillary
% height, area, ...

   C = (VSTR*alphac*Areac.*(h00*gammac - hc)) ./ (gammac - 1);

function [L_Induct] = F_Inductance_ArtVen(D, L, N, mu)
% calculates inductance for the vessel tree

   global BLOOD_DENSITY oParams

   conversion = 980.64;

   Dsize = size(D,1);
   tsize = size(D,2);
   L_Induct = zeros(Dsize, tsize);

   if ((oParams.Womersley == 1) && (abs(oParams.WomersleyTheta) > 0))
      rho = BLOOD_DENSITY;
      omega = 2*pi*oParams.HeartRate;
      theta = (pi/180)*oParams.WomersleyTheta;
      theta_I = pi/2 - theta;
      alpha_I = zeros(Dsize, tsize);
      WRM_I = zeros(Dsize, tsize);
      for k = 1:tsize
         % R(:,k) = (128*mu(:,k).*L(:,k)) ./ (pi*N.*D(:,k).^4);
         alpha_I(:,k) = (D(:,k)/2).*sqrt(rho*omega./(conversion*mu(:,k)));
         WRM_I(:,k) = (1 - (2*besselj(1,j^(3/2)*alpha_I(:,k))) ./ ...
            (j^(3/2).*alpha_I(:,k).*besselj(0,j^(3/2)*alpha_I(:,k)))) ./ ...
            (cos(theta_I) + j*sin(theta_I));
%          L_Induct(:,k) =
real(((4*rho*L(:,k))./(N.*pi.*D(:,k).^2)).*(cos(theta_I)./WRM_I(:,k)));
```

```
            L_Induct(:,k) = (4*BLOOD_DENSITY*L(:,k)) ./
(conversion*pi*N.*D(:,k).^2);
        end
    else
        for k = 1:tsize
            L_Induct(:,k) = (4*BLOOD_DENSITY*L(:,k)) ./
(conversion*pi*N.*D(:,k).^2);
        end
    end

function [L] = F_Inductance_Cap()

    global FUNG_CAPSHEET_SEGMENTS
    L = zeros(FUNG_CAPSHEET_SEGMENTS, 1);


function [V_Ptp, V_Ptp_ref] = F_Model_AirVolume(Ptp, varargin)
% calculates the volume of the lungs as a function of Ptp
% optional variables - morphometry, inflation/deflation

    global oParams
    global MORPHOMETRY_RAT MORPHOMETRY_DOG
MORPHOMETRY_MOUSE
    global MDL_LUNGAIRVOL_INFLATION
MDL_LUNGAIRVOL_DEFLATION
    global CONSTS_MDL_LUNGAIRVOL

    if numel(varargin) > 0
        Morphometry = varargin{1};
        mdlLungAirVolume = varargin{2};
    else
        Morphometry = oParams.Morphometry;
        mdlLungAirVolume = oParams.System_LungAirVolume;
    end

    % choose species
    switch Morphometry
        case MORPHOMETRY_RAT
            oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.RAT;
        case MORPHOMETRY_DOG
            oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.DOG;
        case MORPHOMETRY_MOUSE
            oConstsLungAirVol = CONSTS_MDL_LUNGAIRVOL.MOUSE;
    end
```

```
    k1 = oConstsLungAirVol.k1;
    k2 = oConstsLungAirVol.k2;
    Vm = oConstsLungAirVol.Vm;
    Ptp_FRC = oConstsLungAirVol.Ptp_FRC;

    % choose model
    switch mdlLungAirVolume
       case MDL_LUNGAIRVOL_INFLATION
          Ptp_i = CONSTS_MDL_LUNGAIRVOL.Ptp_i;
          Ptp_e = CONSTS_MDL_LUNGAIRVOL.Ptp_e;
          V_Ptp_i = CONSTS_MDL_LUNGAIRVOL.V_Ptp_i;
          V_Ptp_e = CONSTS_MDL_LUNGAIRVOL.V_Ptp_e;

          V_Ptp = Vm*(((V_Ptp_e-V_Ptp_i)/(Ptp_e-Ptp_i))*(Ptp-Ptp_e) + ...
             V_Ptp_e);
       case MDL_LUNGAIRVOL_DEFLATION
          V_Ptp = Vm * ((1 - k1) * (1 - exp(-k2*Ptp)) + k1);
    end

    V_Ptp_ref = Vm * ((1 - k1) * (1 - exp(-k2*Ptp_FRC)) + k1);

function [hc] = F_Model_CapSheet_Height(Ptm, Ptp)
% solves for capillary height as a fcn of pressure

    global oParams
    global MDL_CAPHEIGHTVSPTM_LIN
MDL_CAPHEIGHTVSPTM_NONLIN1
    global MDL_CAPHEIGHTVSPTM_NONLIN2

    global CONSTS_MDL_CAPHEIGHTVSPTM_LIN
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1
    global CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2

    switch oParams.Cap_Distension
       case MDL_CAPHEIGHTVSPTM_LIN
          oConstsCapHeight = CONSTS_MDL_CAPHEIGHTVSPTM_LIN;
       case MDL_CAPHEIGHTVSPTM_NONLIN1
          oConstsCapHeight = CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN1;
       case MDL_CAPHEIGHTVSPTM_NONLIN2
          oConstsCapHeight = CONSTS_MDL_CAPHEIGHTVSPTM_NONLIN2;
    end

    k1 = oConstsCapHeight.k1;
    h0inf = oConstsCapHeight.h0inf;
    h00 = oConstsCapHeight.h00;
```

```
        alpha = oConstsCapHeight.alpha;
        gamma = oConstsCapHeight.gamma;

        clear oConstsCapHeight

        Psize = size(Ptm,1); tsize = size(Ptp, 2);
        hc = zeros(Psize, tsize);

        switch oParams.Cap_Distension
          case MDL_CAPHEIGHTVSPTM_LIN
            for k = 1:tsize
              hc(:,k) = h00 - (-alpha*Ptm(:,k)/(gamma-1));
            end
          case
{MDL_CAPHEIGHTVSPTM_NONLIN1,MDL_CAPHEIGHTVSPTM_NONLIN2}
            for k = 1:tsize
              hc(:,k) = ((h00 - h0inf)*exp(k1*Ptp(k)) + h0inf) * ...
              (gamma - (gamma-1)*exp((-alpha*Ptm(:,k))/(gamma-1)));
            end

        end

      function [Lc, Wc] = F_Model_CapSheet_Length_Width(Lc_ref, Wc_ref, V_Ptp,
      V_Ptp_ref)
      % determines capillary dimensions as a function of volume

        global oParams
        global MDL_CAPLENWIDVSVOL_ISOTROPIC
      MDL_CAPLENWIDVSVOL_SMITHMITZNER
        global CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER

        Lsize = size(Lc_ref,1); tsize = size(V_Ptp,2);

        switch oParams.Cap_LengthWidthVsVolume
          case MDL_CAPLENWIDVSVOL_ISOTROPIC
            for k = 1:tsize
              Lc(:,k) = Lc_ref .* (V_Ptp(k) / V_Ptp_ref).^(1/3);
              Wc(:,k) = Wc_ref .* (V_Ptp(k) / V_Ptp_ref).^(1/3);
            end
          case MDL_CAPLENWIDVSVOL_SMITHMITZNER
            k1 = CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER.k1;
            k2 = CONSTS_MDL_CAPLENWIDVSVOL_SMITHMITZNER.k2;
            for k = 1:tsize
              Lc(:,k) = Lc_ref .* ((1 - k1)*(V_Ptp(k)/V_Ptp_ref).^k2 + k1);
              Wc(:,k) = Wc_ref .* ((1 - k1)*(V_Ptp(k)/V_Ptp_ref).^k2 + k1);
```

```
            end
        end

    function [epsilonc] = F_Model_CapSheet_PostDiam(epsilonc_ref, ...
        V_Ptp, V_Ptp_ref, hc, hc0FRC)
    % solves for diameter of "posts" in sheet-post capillary bed as fcn of
    % volume.

        global oParams

        global FUNG_CAPSHEET_SEGMENTS
        global MDL_POSTDIAMVSVOL_ISOTROPIC
    MDL_POSTDIAMVSVOL_GILBASED
        global MDL_POSTDIAMVSVOL_CONSTVOL
        global CONSTS_MDL_POSTDIAMVSVOL_GILBASED

        switch oParams.Cap_PostDiamVsVolume
            case MDL_POSTDIAMVSVOL_ISOTROPIC
                epsilonc = epsilonc_ref * ((V_Ptp./V_Ptp_ref).^(2/3)) ...
                    * ones(length(hc), 1);
            case MDL_POSTDIAMVSVOL_GILBASED
                k1 = CONSTS_MDL_POSTDIAMVSVOL_GILBASED.k1;
                k2 = CONSTS_MDL_POSTDIAMVSVOL_GILBASED.k2;
                epsilonc = epsilonc_ref * ((1-k1)*(V_Ptp./V_Ptp_ref).^k2 + k1) ...
                    * ones(length(hc), 1);
            case MDL_POSTDIAMVSVOL_CONSTVOL
                epsilonc = epsilonc_ref * (hc0FRC ./ hc).^(1/2);
        end


    function [Q] = F_Model_CardiacCycle(t, Qmean)

        global oParams
        global MDL_CARDIACCYCLE_SINUSOIDAL
MDL_CARDIACCYCLE_RECTSINE


        HR = oParams.HeartRate;
        switch oParams.System_CardiacCycle
            case MDL_CARDIACCYCLE_SINUSOIDAL
                Q = Qmean*(1+sin(2*pi*HR*t));

              % Q = Qmean*(1+square(1*pi*HR*t));
            case MDL_CARDIACCYCLE_RECTSINE
```

```
            %  Q = Qmean*(1+square(1*pi*HR*t));
              SysDiastRatio = oParams.SysDiasRatio;

              %ts = floor(t * HR)/HR;
              tnew = t - ts;
              phase = (SysDiastRatio/(2*pi*HR))*asin(SysDiastRatio/pi);

              tnew = mod(t+phase, 1/HR);
              coef = (tnew <= (SysDiastRatio/(2*HR)));

              Q = (Qmean / SysDiastRatio) * pi * sin(2*pi*HR*tnew/SysDiastRatio) .*
coef;

          end

        function [D] = F_Model_Diam(D0, Ptm, alpha, gamma)
        % functional model to calculate Diameter D as a function of
        % transmural pressure Ptm

          global oParams
          global MDL_DIAMVSPTM_LIN MDL_DIAMVSPTM_NONLIN

          Psize = size(Ptm,1); tsize = size(Ptm, 2);
          D = zeros(Psize, tsize);

          switch oParams.ArtVen_Distension
            case MDL_DIAMVSPTM_LIN
              for k = 1:tsize
                D(:,k) = D0 .* (1 + alpha .* Ptm(:,k));
              end
            case MDL_DIAMVSPTM_NONLIN
              for k = 1:tsize
                D(:,k) = D0 .* (gamma - (gamma-1).*exp((-alpha.*Ptm(:,k))./(gamma-
1)));
              end
          end


        function [f] = F_Model_FrictionFactor(h, VSTR, epsilon)
        % finds geometric friction factor as function of VSTR, h, epsilon

          global oParams
          global MDL_GEOMFRICTIONFACTOR_FACTOR
          global CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR
```

```
        switch oParams.Cap_GeometricFrictionFactor
            case MDL_GEOMFRICTIONFACTOR_FACTOR
                k1 = CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k1;
                k2 = CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k2;
                k3 = CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k3;
                k4 = CONSTS_MDL_GEOMFRICTIONFACTOR_FACTOR.k4;
        end

        f = (k1*VSTR + k2) * exp((VSTR*k3 + k4) * (h./epsilon));

    function [L] = F_Model_Length(L0, V_Ptp, V_Ptp_ref)
    % determines the length of the vessels based on lung volume

        global oParams
        global MDL_LENVSVOL_ISOTROPIC
MDL_LENVSVOL_SMITHMITZNER
        global CONSTS_MDL_LENVSVOL

        Psize = size(L0, 1); tsize = size(V_Ptp, 2);
        L = zeros(Psize, tsize);

        switch oParams.ArtVen_LengthVsVolume
            case MDL_LENVSVOL_ISOTROPIC
                for k = 1:tsize
                    L(:,k) = L0 .* (V_Ptp(k)/V_Ptp_ref).^(1/3);
                end
            case MDL_LENVSVOL_SMITHMITZNER
                k1 = CONSTS_MDL_LENVSVOL.SMITHMITZNER_k1;
                k2 = CONSTS_MDL_LENVSVOL.SMITHMITZNER_k2;
                for k = 1:tsize
                    L(:,k) = L0 .* (V_Ptp(k)/V_Ptp_ref +
    k1*exp(k2*V_Ptp(k)/V_Ptp_ref)).^(1/3);
                end
        end

    function [Px] = F_Model_PxHat(P, Ptp, Ppl, VesselType)
    % functional models for perivascular pressure Px^

        global oParams
        global CONSTS_MDL_PERIVASCULAR
        global VESSEL_TYPE_ART VESSEL_TYPE_CAP VESSEL_TYPE_VEN
        global MDL_PERIVASCULAR_ALBERT
        global MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER
        global MDL_PERIVASCULAR_BSHOUTY_LAIFOOK
        global MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER
```

```
global MDL_PERIVASCULAR_HAWORTH_LAIFOOK

%Ptp = PA - Ppl;
Psize = size(P, 1);
tsize = size(Ptp, 2);
Px = zeros(Psize, tsize);

%%
switch oParams.ArtVen_PerivascularPressure
   case MDL_PERIVASCULAR_ALBERT
      for k = 1:tsize
         Px(:,k) = -Ptp(k)*ones(Psize,1);
      end
   case MDL_PERIVASCULAR_BSHOUTY_SMITHMITZNER
      switch VesselType
         case VESSEL_TYPE_ART
            ki =
CONSTS_MDL_PERIVASCULAR.BSHOUTY_SMITHMITZNER_ART_ki;
         case VESSEL_TYPE_VEN
            ki =
CONSTS_MDL_PERIVASCULAR.BSHOUTY_SMITHMITZNER_VEN_ki;
      end

      for k = 1:tsize
         Px(:,k) = (ki(1)+ki(2)*Ptp(k)) + (ki(3) + ki(4)*Ptp(k)) * ...
            exp((ki(5)+ki(6)*Ptp(k))*(P(:,k)-Ppl));
      end

   case MDL_PERIVASCULAR_BSHOUTY_LAIFOOK
      switch VesselType
         case VESSEL_TYPE_ART
            ki =
CONSTS_MDL_PERIVASCULAR.BSHOUTY_LAIFOOK_ART_ki;
         case VESSEL_TYPE_VEN
            ki =
CONSTS_MDL_PERIVASCULAR.BSHOUTY_LAIFOOK_VEN_ki;
      end
      %Px = (ki(1)+ki(2)*Ptp) + (ki(3) + ki(4)*Ptp) * ...
      %    exp((ki(5)+ki(6)*Ptp)*(P-Ppl));

      for k = 1:tsize
         Px(:,k) = (ki(1)+ki(2)*Ptp(k)) + (ki(3) + ki(4)*Ptp(k)) * ...
            exp((ki(5)+ki(6)*Ptp(k))*(P(:,k)-Ppl));
      end
```

```
      case MDL_PERIVASCULAR_HAWORTH_SMITHMITZNER
         switch VesselType
            case VESSEL_TYPE_ART
               ki =
CONSTS_MDL_PERIVASCULAR.HAWORTH_SMITHMITZNER_ART_ki;
            case VESSEL_TYPE_VEN
               ki =
CONSTS_MDL_PERIVASCULAR.HAWORTH_SMITHMITZNER_VEN_ki;
         end
         %Px = ((P-Ppl)*ki(3) + ki(4))*Ptp - ((P-Ppl)*ki(3) + ki(4))*ki(2) + ki(1);

         for k = 1:tsize
            Px(:,k) = ((P(:,k)-Ppl)*ki(3) + ki(4))*Ptp(k) - ...
               ((P(:,k)-Ppl)*ki(3) + ki(4))*ki(2) + ki(1);
         end

      case MDL_PERIVASCULAR_HAWORTH_LAIFOOK
         switch VesselType
            case VESSEL_TYPE_ART
               ki =
CONSTS_MDL_PERIVASCULAR.HAWORTH_LAIFOOK_ART_ki;
            case VESSEL_TYPE_VEN
               ki =
CONSTS_MDL_PERIVASCULAR.HAWORTH_LAIFOOK_VEN_ki;
         end
         %Px = ((P-Ppl)*ki(3) + ki(4))*Ptp - ((P-Ppl)*ki(3) + ki(4))*ki(2) + ki(1);

         for k = 1:tsize
            Px(:,k) = ((P(:,k)-Ppl)*ki(3) + ki(4))*Ptp(k) - ...
               ((P(:,k)-Ppl)*ki(3) + ki(4))*ki(2) + ki(1);
         end
      otherwise
         Px = -Ptp*ones(Psize,tsize);
   end


function [mu_a] = F_Model_Viscosity(D, Hctf)
% returns the apparent viscosity as function of D

   global oParams
   global MDL_VISCOSITY_LINEHAN MDL_VISCOSITY_KIANIHUDETZ
   global CONSTS_MDL_VISCOSITY

   D = D*1e4; % convert from cm to um
```

```
switch oParams.System_Viscosity

    case MDL_VISCOSITY_LINEHAN
        % mu_p - plasma viscosity
        mu_p = CONSTS_MDL_VISCOSITY.LINEHAN.mu_p;

        % HctD - hematocrit as a function of diameter
        k1 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k1;
        k2 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k2;
        k3 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k3;
        k4 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k4;
        HctD = Hctf * (k1*exp(-k2*D) + (k4*D./(k3+D)));

        % mu_a - apparent viscosity
        k1 = CONSTS_MDL_VISCOSITY.LINEHAN.mua_k1;
        mu_a = mu_p * exp(k1*HctD);

    case MDL_VISCOSITY_KIANIHUDETZ
        % mu_p - plasma viscosity
        mu_p = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.mu_p;

        % delta - marginal plasma thickness layer
        k1 = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.delta_k1;
        k2 = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.delta_k2;
        delta = k1 - k2 * Hctf;

        % mu_c - blood viscosity in large vessels
        % k1 = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.muc_k1;
        % k2 = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.muc_k2;
        % mu_c = mu_p * exp(k1 + Hctf * k2); % haworth
        % mu_c = exp(k1 + k2*Hctf);

        % using Linehan model to find mu_c
        % HctD - hematocrit as a function of diameter
        k1 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k1;
        k2 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k2;
        k3 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k3;
        k4 = CONSTS_MDL_VISCOSITY.LINEHAN.HctD_k4;
        HctD = Hctf * (k1*exp(-k2*D) + (k4*D./(k3+D)));

        % mu_a - apparent viscosity
        k1 = CONSTS_MDL_VISCOSITY.LINEHAN.mua_k1;
        mu_c = mu_p * exp(k1*HctD);

        % mu_a - apparent viscosity
```

```
        Dmin = CONSTS_MDL_VISCOSITY.KIANIHUDETZ.Dmin;
        D(find(D<Dmin)) = Dmin;
        mu_a =  mu_p * (1 - (1 - mu_p./mu_c).*(1 - (2*delta)./D).^4).^(-1) .* ...
           (1 - (Dmin./D).^4).^(-1);
    end

function [R] = F_Resistance_ArtVen(D, L, N, mu)
% calculates resistance based on diameter, viscosity
    global BLOOD_DENSITY oParams

    conversion = 980.64;

    Dsize = size(D,1);
    tsize = size(D,2);
    R = zeros(Dsize, tsize);

    if ((oParams.Womersley == 1) && (abs(oParams.WomersleyTheta) > 0))
        rho = BLOOD_DENSITY;
        omega = 2*pi*oParams.HeartRate;
        theta = (pi/180)*oParams.WomersleyTheta;
        theta_R = theta;
        alpha_R = zeros(Dsize, tsize);
        WRM_R = zeros(Dsize, tsize);
        for k = 1:tsize
            % R(:,k) = (128*mu(:,k).*L(:,k)) ./ (pi*N.*D(:,k).^4);
            alpha_R(:,k) = (D(:,k)/2).*sqrt(rho*omega./(conversion*mu(:,k)));
            WRM_R(:,k) = (1 - (2*besselj(1,j^(3/2)*alpha_R(:,k))) ./ ...
               (j^(3/2).*alpha_R(:,k).*besselj(0,j^(3/2)*alpha_R(:,k)))) ./ ...
               (cos(theta_R) + j*sin(theta_R));
            R(:,k) = real(((128.*mu(:,k).*L(:,k))./(pi.*N.*D(:,k).^4)) .* ...
               ((alpha_R(:,k).^2*sin(theta_R)) ./ (8*WRM_R(:,k))));
        end
    else
        for k = 1:tsize
            R(:,k) = (128*mu(:,k).*L(:,k)) ./ (pi*N.*D(:,k).^4);
        end
    end

function [R] = F_Resistance_Cap(hc, Lc, Area, fc, mu, VSTR)
% capillary resistance
% Zone 3
    R = (12*mu.*fc.*(Lc.^2))./(Area.*(hc.^3)*VSTR);

function [V] = F_Volume_ArtVen(D, L, N)
% returns the volume of a vessel as function of diameter, length
```

```
    Dsize = size(D,1); tsize = size(D,2);
    V = zeros(Dsize, tsize);

    for k = 1:tsize
       V(:,k) = (pi*(D(:,k).^2).*L(:,k).*N)/4;
    end
function [V] = F_Volume_Cap(hc, Area, VSTR)
% returns the volume of capillary sheet as function of thickness, area

    V = Area .* hc * VSTR;
```

3. Folder Result

```
function [status] = ExcelExport(gSim_Results)
%global gSim_Results

% Number of Simulations

Number = length(gSim_Results);
% Check if the option is SS or Dynamic
%for i = 1:Number
%% Adding Simulation Paramters
Parameterlist = 'Simulation Parameters';
% Making array of labels for Parameters
Funclist = 'Functional Models';
SSname = 'Steady State Simulation Results';
DYname = 'Dynamic Simulation Results';

Sim_Name =
(horzcat({gSim_Results{1,Number}.Params.SimulationName(:,:)}));

Parameters = {'Simulation Parameters'
   'SimulationName';...
   'SimulationType';...
   'Morphometry';...
   'OpenFile';...
   'Area';...
   'VSTR';...
   'Mean Flow(ml/sec)';...
   'Pv(cm-H20)';...
    'PA(cm-H20)';...
     'Ppl(cm-H20)';...
      'Hct(ratio)';...
   'TimeStep';...
```

```
            'RunTime';...
            'HeartRate';...
            'SysDiasRatio';...
            'Breathing Rate';...
            'Wall Resistance';...
            'Viscoelastic Coefficient Ratio';...
            'Viscoelasiticity Threshold';...
            'LeadInTime';...
            'MinPtp';...
            'MaxPtp';...
            'Womersley';...
            'VariableRLC';...
            'ArtVaso';...
            'ArtBeta1';...
            'ArtDRatio';...
            'VenVaso';...
            'VenBeta1';...
            'VenDRatio';...
            'VenAlphaRatio'};

        s = gSim_Results{1,Number}.Params.SimulationType(1,1);

        if s == 1
            simtype = 'Steady State Simulation';
        else
            simtype = 'Dynamic Simulation';
        end
        % Ordering Results
          Parameters_result =
(horzcat({gSim_Results{1,Number}.Params.SimulationName(:,:),...
              simtype,...
            gSim_Results{1,Number}.Params.Morphometry(1,1)',...
            gSim_Results{1,Number}.Params.OpenFile(:,:),....
            gSim_Results{1,Number}.Params.Area(:,1)',...
            gSim_Results{1,Number}.Params.VSTR(1,1)',...
            gSim_Results{1,Number}.Params.Q(1,1)',...
            gSim_Results{1,Number}.Params.Pv(1,1)',...
            gSim_Results{1,Number}.Params.PA(1,1)',...
            gSim_Results{1,Number}.Params.Ppl(1,1)',...
            gSim_Results{1,Number}.Params.Hct(1,1)',...
            gSim_Results{1,Number}.Params.TimeStep(1,1)',...
            gSim_Results{1,Number}.Params.RunTime(1,1)',...
            gSim_Results{1,Number}.Params.HeartRate(1,1)',...
            gSim_Results{1,Number}.Params.SysDiasRatio(1,1)',...
            gSim_Results{1,Number}.Params.BreathingRate(1,1)',...
```

```
        gSim_Results{1,Number}.Params.WallResistanceX(1,1)',...
        gSim_Results{1,Number}.Params.ViscoCoefRatio(1,1)',...
        gSim_Results{1,Number}.Params.ViscoThreshold(1,1)',...
        gSim_Results{1,Number}.Params.WomersleyTheta(1,1)',...
        gSim_Results{1,Number}.Params.LeadInTime(1,1)',...
        gSim_Results{1,Number}.Params.MinPtp(1,1)',...
        gSim_Results{1,Number}.Params.MaxPtp(1,1)',...
        gSim_Results{1,Number}.Params.ArtVaso(1,1)',...
        gSim_Results{1,Number}.Params.ArtBeta1(1,1)',...
        gSim_Results{1,Number}.Params.ArtDRatio(1,1)',...
        gSim_Results{1,Number}.Params.ArtAlphaRatio(1,1)',...
        gSim_Results{1,Number}.Params.VenVaso(1,1)',...
        gSim_Results{1,Number}.Params.VenBeta1(1,1)',...
        gSim_Results{1,Number}.Params.VenDRatio(1,1)',...
        gSim_Results{1,Number}.Params.VenAlphaRatio(1,1)'}))';

    % Select xls file to write to
    [xlsFileName,xlsPathName] = uigetfile('*.xls*','Select the xls file to write results
into ', 'MultiSelect','off');
    xlsFile = fullfile(xlsPathName,xlsFileName);

    % writing title/header labels
    commandwindow
    display(['Writing results into file ' xlsFile ])


    warning off MATLAB:xlswrite:AddSheet
    [status message] = xlswrite(xlsFile, Parameters,[Parameterlist],'A1');

    % writing data results

    [status message] = xlswrite(xlsFile, Parameters_result, [Parameterlist], 'B2');

    switch status
      case 1
        display('Results: File write successful')
        display(message.message)
      case 0
        display('Results: File write UNSUCCESSFUL')
        display(message.message)
        display(message.identifier)
    end

    %% Adding Functional Models
```

```
FunctionalModels_List = {'ArtVen_Distension';...
'Cap_Distension';...
'ArtVen_PerivascularPressure';...
'System_Viscosity' ;...
'System_LungAirVolume' ;...
'ArtVen_LengthVsVolume';...
'Cap_LengthWidthVsVolume';...
'Cap_PostDiamVsVolume' ;...
'Cap_Zone2Geometry' ;...
'System_CardiacCycle';...
'System_BreathingCycle' ;...
'System_HctFahreusEffect';...
'Cap_GeometricFrictionFactor'};

Func_result =
(horzcat({gSim_Results{1,Number}.Params.ArtVen_Distension(1,1)',...
    gSim_Results{1,Number}.Params.Cap_Distension(1,1)',....
    gSim_Results{1,Number}.Params.ArtVen_PerivascularPressure(1,1)',....
    gSim_Results{1,Number}.Params.System_Viscosity(:,1)',....
    gSim_Results{1,Number}.Params.System_LungAirVolume(:,1)',...
    gSim_Results{1,Number}.Params.ArtVen_LengthVsVolume(1,1)',...
    gSim_Results{1,Number}.Params.Cap_LengthWidthVsVolume(1,1)',...
    gSim_Results{1,Number}.Params.Cap_PostDiamVsVolume(1,1)',...
     gSim_Results{1,Number}.Params.Cap_Zone2Geometry(1,1)',...
    gSim_Results{1,Number}.Params.System_CardiacCycle(1,1)',...
    gSim_Results{1,Number}.Params.System_BreathingCycle(:,1)',...
    gSim_Results{1,Number}.Params.System_HctFahreusEffect(1,1)',...
    gSim_Results{1,Number}.Params.Cap_GeometricFrictionFactor(1,1)'}))';

  warning off MATLAB:xlswrite:AddSheet
[status message] = xlswrite(xlsFile, {'Functional Models'}, [Funclist], 'A1');
switch status
   case 1
      display('Header: File write successful')
      display(message.message)
   case 0
      display('Header: File write UNSUCCESSFUL')
      display(message.message)
      display(message.identifier)
end
% writing row labels

[status message] = xlswrite(xlsFile, FunctionalModels_List, [Funclist],'A3');
switch status
   case 1
```

```
            display('Row Labels: File write successful')
            display(message.message)
        case 0
            display('Row Labels: File write UNSUCCESSFUL')
            display(message.message)
            display(message.identifier)
    end



    % writing data results

    [status message] = xlswrite(xlsFile, Func_result, [Funclist], 'B3');

    switch status
        case 1
            display('Results: File write successful')
            display(message.message)
            status = 1;
        case 0
            display('Results: File write UNSUCCESSFUL')
            display(message.message)
            display(message.identifier)
            status = -1;
    end

    %% Adding Steady State simulation result

    SS_Infolist = {'SimulationName';...
        'SimulationType'};
    SS_Infolist_data =
(horzcat({gSim_Results{1,Number}.Params.SimulationName(:,:),...
        simtype}))';

    SS_List_Art = {'Arteries';...
    'Pressure - Arteries(cm-H20)';...
    'Volume - Arteries(ml)';...
    'Resistance - Arteries(cm-H20sec/ml)' ;...
    'Inertance - Arteries(cm-H20ml-1s2)' ;...
    'Compliance - Arteries(ml/cm-H20)'};
    SS_List_Cap = {
    'Capillaries';...
    'Pressure - Capillaries(cm-H20)';...
    'Volume - Capillaries(ml)';...
    'Resistance - Capillaries(cm-H20sec/ml)' ;...
```

```
'Inertance - Capillaries(cm-H20ml-1s2)' ;...
'Compliance - Capillaries(ml/cm-H20)'};
SS_List_Veins = {
'Veins';...
 'Pressure - Veins(cm-H20)';...
'Volume - Veins(ml)';...
'Resistance - Veins(cm-H20sec/ml)' ;...
'Inertance - Veins(cm-H20ml-1s2)' ;...
'Compliance - Veins(ml/cm-H20)'};
SS_PVsF = {
'Mean Flow (ml/sec)';...
'Pulmonary arterial pressure (cm-H20)'};


SS_result_art = vertcat((gSim_Results{1,Number}.STEADY.Pmidart(:,1))',...
   gSim_Results{1,Number}.STEADY.VOLart(:,1)',...
   gSim_Results{1,Number}.STEADY.Rart(:,1)',...
   gSim_Results{1,Number}.STEADY.Iart(:,1)',...
   gSim_Results{1,Number}.STEADY.Cart(:,1)');

SS_result_cap = vertcat((gSim_Results{1,Number}.STEADY.Pmidcap(:,1))',...
   gSim_Results{1,Number}.STEADY.VOLcap(:,1)',...
   gSim_Results{1,Number}.STEADY.Rcap(:,1)',...
   gSim_Results{1,Number}.STEADY.Icap(:,1)',...
   gSim_Results{1,Number}.STEADY.Ccap(:,1)');

SS_result_ven = vertcat((gSim_Results{1,Number}.STEADY.Pveins(:,1))',...
   gSim_Results{1,Number}.STEADY.VOLvein(:,1)',...
   gSim_Results{1,Number}.STEADY.Rveins(:,1)',...
   gSim_Results{1,Number}.STEADY.Iveins(:,1)',...
   gSim_Results{1,Number}.STEADY.Cveins(:,1)');

SS_PVsF_result = vertcat((gSim_Results{1,1}.MeanFlow(:,:)),...
   gSim_Results{1,1}.PAP(:,:));



 warning off MATLAB:xlswrite:AddSheet

% writing row labels


[status message] = xlswrite(xlsFile,SS_Infolist, [SSname],'A1');
[status message] = xlswrite(xlsFile,SS_Infolist_data, [SSname],'B1');
```

```
[status message] = xlswrite(xlsFile,SS_List_Art, [SSname],'A4');
% writing data results
[status message] = xlswrite(xlsFile, SS_result_art, [SSname], 'B5');

[status message] = xlswrite(xlsFile,SS_List_Cap, [SSname],'A12');
% writing data results
[status message] = xlswrite(xlsFile, SS_result_cap, [SSname], 'B13');

[status message] = xlswrite(xlsFile,SS_List_Veins, [SSname],'A22');
% writing data results
[status message] = xlswrite(xlsFile, SS_result_ven, [SSname], 'B23');

% writing data results
[status message] = xlswrite(xlsFile, SS_PVsF, [SSname], 'A31');
% writing data results
[status message] = xlswrite(xlsFile, SS_PVsF_result, [SSname], 'B31');


switch status
   case 1
      display('Results: File write successful')
      display(message.message)
      status = 1;
   case 0
      display('Results: File write UNSUCCESSFUL')
      display(message.message)
      display(message.identifier)
      status = -1;
end

%% Adding  Dynamic Results
s = gSim_Results{1,Number}.Params.SimulationType(1,1);

if s ==2



DY_Infolist = {'SimulationName';...
   'SimulationType'};
DY_Infolist_data =
(horzcat({gSim_Results{1,Number}.Params.SimulationName(1,1)',...
   simtype}))';

DY_List_Art = {'Arteries';...
```

```
'Pressure - Arteries(cm-H20)';...
};

DY_List_Art_F = {'Flow - Arteries(cm-H20)';...
};

DY_List_Cap = {
'Capillaries';...
'Pressure - Capillaries(cm-H20)';...
 };

 DY_List_Veins = {
 'Veins';...
  'Pressure - Veins(cm-H20)'};

DY_List_Veins_F = {'Veins';...
'Flow - Veins(cm-H20)';...
};


DY_result_art = gSim_Results{1,Number}.DYNAMIC.Pmidart(:,:);
DY_result_art_f = gSim_Results{1,Number}.DYNAMIC.Flowart(:,:);
DY_result_cap = gSim_Results{1,Number}.DYNAMIC.Pmidcap(:,:);
DY_result_ven = gSim_Results{1,Number}.DYNAMIC.Pmidvein(:,:);
DY_result_ven_f = gSim_Results{1,Number}.DYNAMIC.Flowvein(:,:);
  warning off MATLAB:xlswrite:AddSheet

% writing row labels
DY_List_Lab = {'Simulation Type'};
DY_List_Lab2 = {'Dynamic Simulation'};
 [status message] = xlswrite(xlsFile,DY_List_Lab,[DYname],'A1');
 [status message] = xlswrite(xlsFile,DY_List_Lab2,[DYname],'B1');

% Arteries - Pressure and Flow
[status message] = xlswrite(xlsFile,DY_List_Art, [DYname],'A4');

[status message] = xlswrite(xlsFile, DY_result_art, [DYname], 'B5');

[status message] = xlswrite(xlsFile,DY_List_Art_F, [DYname],'A24');

[status message] = xlswrite(xlsFile, DY_result_art_f, [DYname], 'B24');

% Capillaries –Pressure

[status message] = xlswrite(xlsFile,DY_List_Cap, [DYname],'A44');
```

```
[status message] = xlswrite(xlsFile, DY_result_cap, [DYname], 'B45');

% Veins - Pressure and Flow

[status message] = xlswrite(xlsFile,DY_List_Veins, [DYname],'A48');

[status message] = xlswrite(xlsFile, DY_result_ven, [DYname], 'B49');

[status message] = xlswrite(xlsFile,DY_List_Veins_F, [DYname],'A70');

[status message] = xlswrite(xlsFile, DY_result_ven_f, [DYname], 'B71');


switch status
   case 1
      display('Results: File write successful')
      display(message.message)
      status = 1;
   case 0
      display('Results: File write UNSUCCESSFUL')
      display(message.message)
      display(message.identifier)
      status = -1;
end
msgbox(' Steady and Dynamic state simulation file write successful');
else
msgbox(' Steady state simulation_file write successful');
end


function [r] = ModelResults()
% ModelResults - Class to store model results from PC Physiome

r.Params = ModelParams;

r.STEADY.Pnode = [];
r.STEADY.Pmid = [];
r.STEADY.D = [];
r.STEADY.V = [];
r.STEADY.L = [];
r.STEADY.R = [];
r.STEADY.C = [];
r.STEADY.I = [];
r.STEADY.mu = [];
```

```
r.DYNAMIC.t = [];
r.DYNAMIC.Y = [];
r.DYNAMIC.Pmid = [];
r.DYNAMIC.Q = [];
r.DYNAMIC.D = [];
r.DYNAMIC.V = [];
r.DYNAMIC.L = [];
r.DYNAMIC.R = [];
r.DYNAMIC.C = [];
r.DYNAMIC.I = [];
r.DYNAMIC.mu = [];
```